جامعة سيدي محمد بن عبد الله بفاس
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵙⵉⴷⵉ ⵎⵓⵃⴰⵎⵎⴷ ⵏ ⵄⴰⴱⴷⵍⵍⴰⵀ ⴳ ⴼⴰⵙ
UNIVERSITÉ SIDI MOHAMED BEN ABDELLAH DE FES

كلية العلوم ظهر المهراز فاس
ⵜⴰⵖⵢⵍⵉⵏⵜ ⵏ ⵜⵣⵔⴼⵜ ⵎⵣⴷⵓⵔ ⵎⵣⴷⵓⵔ ⵅ ⴼⴰⵙ
FACULTE DES SCIENCES DHAR EL MAHRAZ FES

N° d'ordre : 51/2023

Centre des Etudes Doctorales : Sciences et Technologies de l'Information et de Communication

Formation Doctorale : Informatique

Discipline Informatique

Spécialité Informatique

Laboratoire : Laboratoire d'Informatique, Signaux, Automatique et Cognitivisme LISAC

Thèse pour l'obtention de DOCTORAT

Toward a Better Driving Experience:

A Machine Learning and Model Predictive Control Approach

Préparée par : LAMOUIK Imad

Soutenue le 27/05/2023 devant le jury :

Nom et Prénom	Grade	Etablissement	Qualité
NFAOUI El Habib	PES	Faculté des Sciences Dhar El Mahraz, Fès	Président
EL FALLAHI Abdellah	PES	Ecole Nationale des Sciences Appliquées, Tétouan	Rapporteur & examinateur
TAIRI Hamid	PES	Faculté des Sciences Dhar El Mahraz, Fès	Rapporteur & examinateur
AGHOUTANE Badraddine	PH	Faculté des Sciences, Meknès	Rapporteur & examinateur
GUALOUS Hamid	PES	Université de Caen Normandie, France	Examinateur
RIFFI Jamal	PH	Faculté des Sciences Dhar El Mahraz, Fès	Examinateur
MAHRAZ Mohamed Adnane	PH	Faculté des Sciences Dhar El Mahraz, Fès	Examinateur
SABRI My Abdelouahed	PES	Faculté des Sciences Dhar El Mahraz, Fès	Co-directeur de thèse
YAHYAOUY Ali	PES	Faculté des Sciences Dhar El Mahraz, Fès	Directeur de thèse

Année Universitaire : 2021-2022

Acknowledgements

I would like to take this opportunity to express my deepest gratitude to all those who have supported me throughout my journey to completing this thesis. Their unwavering guidance, encouragement, and assistance have been invaluable to me, and I am truly indebted to them.

First and foremost, I would like to extend my heartfelt appreciation to my thesis director, **Prof. Ali Yahyaouy**, and to my supervisor, **Prof. My Abdelouahed SABRI**. Your profound knowledge, insightful feedback, and continuous support have played a vital role in shaping this research work. I am grateful for your patience, dedication, and willingness to guide me through every step of the way.

I am also immensely thankful to the faculty members of the **Faculty of Sciences Dhar El Mahraz Fez** and the **Center of Doctoral Studies CED-Faculty of Sciences Dhar El Mahraz Fez** for their contributions in shaping my academic journey. Their expertise and passion for teaching have greatly enriched my understanding of the subject matter.

I extend my sincere thanks to the members of my thesis committee, **Prof. NFAOUI El Habib**, **Prof. EL FALLAHI Abdellah**, **Prof. TAIRI Hamid**, **Prof. AGHOUTANE Badraddine**,**Prof. GUALOUS Hamid**, **Prof. RIFFI Jamal** and, **Prof. MAHRAZ Mohamed Adnane** for their valuable time, constructive criticism, and thoughtful suggestions. Their thorough review and insightful comments have undoubtedly enhanced the quality and rigor of this research.

To my beloved family, **Brahim** and **Rachida**, and my three amazing sisters, **Siham**, **Houda**, and **Salma**, I owe the utmost gratitude for your unwavering support, love, and understanding throughout this challenging endeavor. Your belief in me, encouragement, and sacrifices have been the driving force behind my success. I am forever indebted to each and every one of you.

Lastly, I would like to express my heartfelt appreciation to all the participants who willingly shared their time and insights, making this research possible. Their willingness to participate and provide valuable information has contributed immensely to the validity and relevance of this study.

Thank you.

Contents

Contents ii

List of Figures iv

List of Tables vii

List of Acronyms viii

Abstract x

Résumé xii

General Introduction 1
1 Background and motivations 1
2 Advanced traffic control for autonomous vehicle 6
3 Problems formulation 7
4 List of publications & contributions 8

Chapter 1. Overview 10
1 Intoduction 10
2 The problem of optimal traffic routing 11
3 The problem of Vehicle overtaking 28
4 Related Works 32
5 Conclusion 36

Chapter 2. Multi-agent traffic coordinator for A.Vs at intersections 37
1 Introduction 37
2 Preliminaries 38
3 Proposed system 48
4 Results and Discussion 52
5 Conclusion 53

Chapter 3. Multi-Cluster Q-routing for traffic networks **54**
1 Introduction 54
2 Preliminaries 54
3 Proposed Model 57
4 Results and Discussion 59
5 Conclusion 61

Chapter 4. The problem of Vehicle overtaking **63**
1 Introduction 63
2 Overtaking strategies 65
3 Vehicle modeling 66
4 Object detection and tracking 71
5 Model predictive control 77
6 System architecture 81
7 Results and Discussion 81
8 Test scenarios 82
9 Conclusion 88

Conclusion and Future Directions **91**

A. Résumé étendu en français **93**
1 Introduction 93
2 Le problème du routage optimal du trafic 93
3 Le problème du dépassement des véhicules 100

Bibliography **108**

List of Figures

.1 Dataset statistics: grouped by hour. 5
.2 Dataset: grouped by hour. 5
.3 Dataset: grouped by day. 6

2.1 Illustrating the dynamic relationship between agents and their surroundings. . 38
2.2 The linear threshold unit. 39
2.3 Perceptron diagram. 39
2.4 Basic structure of a Multi-Layer Perceptron. 40
2.5 Model complexity and overfitting. 41
2.6 Standard model of reinforcement learning. 43
2.7 Q-learning algorithm 46
2.8 Q-learning 46
2.9 Deep Q-learning 47
2.10 Proposed system Components' overview. 48
2.11 Proposed system algorithm. 49
2.12 Exchange of information between the intersection agent and the vehicle agent. 50
2.13 Vehicles arrangement to input to the deep neural network. 51
2.14 Vehicles arrangement to input to the deep neural network. 51
2.15 Estimated Max Q-value. 52
2.16 Received reward. 52
2.17 Simulation screen-shot. 53

3.1 Example of Q-routing (solid arrows represent vehicles being routed and dashed arrows represent traffic state propagation). 56
3.2 Graph clustering example. 56
3.3 Traffic network clustering. 58
3.4 The routing process diagram. 59
3.5 The city of Fez, Morocco. (a) before clustering and (b) after clustering into 10 distinct clusters. 60
3.6 Average vehicle arrival time under normal traffic conditions. 60
3.7 Average vehicle arrival time under stochastically generated traffic congestion . 61
3.8 Mean routing table size. 61

4.1 The Overtaking maneuver. 66
4.2 Reference Coordinate Frames 67
4.3 Kinematic Bicycle Model. 67
4.4 Dynamic Bicycle Model. 68
4.5 A scheme of the tire forces, notations are for the rear wheel. 70
4.6 Curvilinear coordinate system 71
4.7 YOLO for object detection. 72
4.8 YOLO network architecture. 72
4.9 Structure of the Darknet53 convolutional network. 73
4.10 Yolov4 Object detector. 74
4.11 An illustration of two cameras mounted together as a stereo camera. 75
4.12 Illustration of the angles used for computing the distance. 76
4.13 Structure of MPC. 78
4.14 Basic concept of MPC. 78
4.15 System architecture. 81
4.16 Example a picture captured by the right camera of a vehicle. 83
4.17 Object detection and velocity estimation. 83
4.18 Simulation of a right-side passing maneuver showing a bird's-eye view. 84
4.19 Vehicle'speed profile during the simulation. 84
4.20 Successful passing and safety margin violations based on the speed of the lead vehicle, both with and without fog present. 85
4.21 Bird's-eye view simulation of a right-side passing maneuver involving one vehicle moving in the same direction as the vehicle being simulated, and another incoming vehicle. 85
4.22 Speed of the vehicle being simulated during two scenarios: 1) immediate passing, 2) delayed passing. 86
4.23 Successful passing and safety margin violations based on the speed of the lead vehicle during an immediate passing maneuver. 86
4.24 Successful passing and safety margin violations based on the speed of the lead vehicle during a postponed passing maneuver. 87
4.25 Simulation of a right-side passing maneuver viewed from above on a curved road. 87
4.26 Speed of the vehicle being simulated during two scenarios: 1) immediate passing, 2) delayed passing. 88
4.27 Successful passing and safety margin violations based on the speed of the lead vehicle during an immediate passing maneuver. 88
4.28 Successful passing and safety margin violations based on the speed of the lead vehicle during a delayed passing maneuver. 89
4.29 Successful overtake and safety margin violation when the leading vehicle performs a speed change mid-overtake. 89
4.30 Successful maneuvers with leading vehicle's random speed change during overtaking 90

A.1 Exemple de routage Q (les flèches pleines représentent les véhicules en cours de routage et les flèches en pointillés représentent la propagation de l'état du trafic). 95
A.2 Regroupement du réseau de trafic. 96
A.3 Le schéma du processus de routage. 97

A.4 La ville de Fès, Maroc. (a) avant regroupement et (b) après regroupement en 10 groupes distincts. 97
A.5 Temps moyenne d'arrivée des véhicules dans des conditions normales de circulation. 98
A.6 Temps d'arrivée moyen des véhicules dans des embouteillages générés de manière stochastique. 98
A.7 Taille moyenne de la table de routage. 99
A.8 Notation du modèle de vélo 102
A.9 Système de coordonnées curvilignes 104
A.10 System architecture. 106

List of Tables

.1 2019 Fatal Crash Reports for Drivers & Motorcycle Operators. 3
.2 Classifying vehicles based on their length. 4
.3 An example of data from a generated file for a two-lane road with vehicle classifications. 4
.4 Dataset sample. 4

4.1 Notations. 69
4.2 Hyperparameters values. 82
4.3 Object tracking performance. 83

A.1 Notations. 102

List of Acronyms

AODV Ad-hoc On-demand Distance Vector

CNN Convolutional neural network

COG Center of Gravity

DNN Deep neural network

DQN Deep Q Learning

DSR Dynamic Source Routing

FPN Feature pyramid network

IA Intersection agent

ICRA Inner Cluster Routing Agents

ITS Intelligent transportation systems

LiDAR Light Detection and Ranging

MAS Multi-agent systems

MDP Markov decision process

MIMO Multi-input multi-output

MODA Multiple Object Detection Accuracy

MODP Multiple Object Detection Precision

MOTA Multiple object tracking accuracy

MOTP Multiple objects tracking precision

MPC Model predictive control

OCRA Outer Cluster Routing Agent

R-CNN Region Based Convolutional Neural Networks

ReLU Rectified Linear Unit

RPN Region proposal network

SAM Segment Anything Model

SSD Single Shot Detector

TD Temporal-Difference

VA Vehicle agent

YOLO You only look once

Abstract

Traffic control is a topic of significant interest to researchers around the world, and has been for the past century. The need to improve the driving experience, reduce congestion, and eliminate traffic accidents has significantly driven research in this field. And with the rapid advance of computing power and technology, it is now possible to develop sophisticated control solutions that are capable of handling a wide range of complex traffic tasks, making autonomous driving systems a more viable solution. In short, traffic control research is essential for addressing the challenges of modern transportation systems, and the recent technological advancements in this field have opened up new possibilities for improving the safety and efficiency of our roads.

Traffic congestion is a major problem that affects cities around the world, causing longer commutes and increased pollution. In this thesis, we sought to address this issue by exploring the potential for optimal traffic routing to improve the driving experience. We recognized that traffic congestion has become a daily annoyance for many drivers and that finding a solution to this problem is crucial for enhancing the quality of life in modern cities.

In addition, we recognized that the development of autonomous vehicles requires a reliable and efficient routing strategy to calculate optimal paths. However, we found that existing static and centralized algorithms are often limited in their ability to solve the congestion problem, especially when considering the size and complexity of modern cities and the volume of daily traffic. Therefore, we focused on exploring alternative approaches to traffic routing that are better suited to addressing these challenges.

To address the limitations of existing traffic routing algorithms, we proposed two innovative systems in this thesis. The first system utilizes the latest advances in artificial intelligence, particularly deep neural networks, to train models that are capable of routing traffic through dynamic traffic grids. The second system combines the power of Q-routing with spectral clustering to offer a scalable routing solution that is suitable for use in large traffic grids.

As a result of these proposed approaches, we were able to develop routing systems that are highly efficient and able to optimize resource utilization while minimizing average trip length, travel time, and vehicle fuel consumption. These systems represent a significant

improvement over traditional traffic routing algorithms and have the potential to greatly improve the driving experience.

On the other side, overtaking is a complex and potentially dangerous maneuver that is frequently performed by drivers, and it presents a significant challenge for autonomous systems. Each year, numerous fatal accidents are caused by improper overtaking, and the task of evaluating the safety of passing a leading vehicle on a two-lane road is particularly challenging. Despite this, few realistic attempts have been made in the literature to provide practical solutions for this problem, with many proposed approaches relying on simplifications of vehicle models and assumptions about the state of the environment that make them unsuitable for use in real-world scenarios.

To address this issue, we proposed an architecture based on model predictive control that is capable of performing safe overtaking in the presence of conflicting traffic. Our system is able to accurately estimate the safety of starting an overtake maneuver, as well as control the vehicle during the maneuver to ensure a collision-free overtake. This is achieved through the use of a complete and realistic model of the vehicle's dynamics, including a dynamic representation of the vehicle's tire forces, which allows for robust and safe driving.

In addition, our system employs machine-learning techniques, including Yolo and Deepsort, to gather information about the environment through the use of stereoscopic vision. This allows us to gather data on factors such as lane width, lane center, and the distance from neighboring vehicles, which is essential for accurately evaluating the safety of an overtake maneuver. Overall, our system represents a major advancement in the field of autonomous overtaking and has the potential to significantly reduce the risk of vehicle crashes during high-speed maneuvers.

Keywords: Traffic control, Autonomous driving, Q-routing, Overtaking, Congestion, Optimal traffic routing, Traffic safety.

Résumé

Le contrôle de la circulation est, depuis le siècle dernier, à l'épicentre de l'intérêt des chercheurs, encouragé par le besoin émergent d'apporter des améliorations à l'expérience de conduite, à la réduction des embouteillages et à l'élimination des accidents de la circulation. Et en raison de l'augmentation de la puissance de calcul et des progrès technologiques, il est devenu possible de développer des solutions de contrôle plus robustes capables de gérer diverses tâches complexes de trafic qui étaient autrement loin de la portée des systèmes de conduite autonomes.

Les embouteillages sont connus pour entraîner des trajets plus longs et plus de pollution, ainsi, en offrant une solution de routage optimale, nous visons à améliorer l'expérience de conduite. Par conséquent, dans cette thèse, nous avons commencé par considérer l'acheminement optimal du trafic comme une solution au problème de congestion du trafic puisqu'il est devenu une gêne quotidienne pour les conducteurs dans la plupart des villes modernes.

De plus, afin d'atteindre une autonomie totale, une stratégie de routage fiable et rapide est nécessaire pour calculer les chemins optimaux, mais les algorithmes statiques et centralisés existants ont d'énormes limites qui les empêchent de résoudre le problème de la congestion, surtout si l'on considère la taille des villes modernes et le volume du trafic quotidien.

Par conséquent, le premier système proposé tire parti des avancées de l'intelligence artificielle, en particulier des réseaux de neurones profonds, pour former des modèles capables d'acheminer le trafic à travers des grilles de trafic dynamiques. La seconde utilise la puissance du Q-routing combinée au clustering Spectral pour offrir une solution de routage plus évolutive qui peut être utilisée dans les grands réseaux de trafic. Par conséquent, les systèmes de routage proposés sont capables d'un routage efficace qui optimise l'utilisation des ressources et minimise la longueur moyenne du trajet, le temps de trajet et la consommation de carburant du véhicule.

Deuxièmement, nous avons abordé le problème du dépassement des véhicules, comme l'une des manœuvres les plus dangereuses effectuées par les conducteurs et aussi l'une des manœuvres les plus difficiles pour les systèmes autonomes, qui entraîne d'innombrables accidents mortels chaque année. En particulier, comment évaluer la sécurité du dépasse-

ment efficace et fiable d'un véhicule de tête sur une route à deux voies. Cependant, peu de tentatives réalistes dans ce domaine ont été faites dans la littérature pour apporter des solutions pratiques sans connaissance préalable de l'état de l'environnement et des simplifications des modèles de véhicules. Ces simplifications de modèles rendent de nombreuses solutions proposées dans la littérature inutilisables dans des scénarios réels. Et compte tenu des dangers qui peuvent découler d'un dépassement défectueux et du risque important d'accidents de véhicules lors de manœuvres à grande vitesse.

Par conséquent, nous avons proposé une architecture basée sur un modèle de contrôle prédictif qui produit capable d'effectuer un dépassement en toute sécurité d'un véhicule de tête en présence de trafic conflictuel. Le système peut estimer avec précision la sécurité de démarrage d'un dépassement du véhicule en plus du contrôle du véhicule pendant la manœuvre qui garantit un dépassement sans collision en utilisant un modèle complet et réaliste de la dynamique du véhicule. Le système s'appuie sur une représentation dynamique de la dynamique du véhicule ainsi que des forces des pneus pour produire un modèle fidèle de l'interaction du véhicule avec la route résultant en une conduite robuste et sûre. Le système utilise une approche de vision stéréoscopique et des techniques d'apprentissage automatique, principalement Yolo et Deepsort, pour recueillir des informations sur l'environnement telles que la largeur de la voie, le centre de la voie et la distance par rapport aux véhicules voisins.

Mots clés: Contrôle de la circulation, Conduite autonome, Dépassement, Congestion, Acheminement optimal du trafic, Sécurité routière.

General Introduction

1 Background and motivations

1.1 Traffic congestion

According to a 2019[1] study, drivers spend about 18 days per year driving or an average of more than eight hours per week. This number is likely to continue increasing due to **longer daily commutes** and **increased traffic congestion**. The rise in congestion can be attributed to various factors, including a larger number of vehicles on the road as a result of more affordable prices and a growing middle-class population, as well as outdated infrastructure that was not designed to handle the current level of traffic demand. Most roads also still rely on inadequate traffic signaling systems, particularly intersection traffic lights, which use static time intervals to switch between red and green, regardless of the current traffic conditions. These static signals can also contribute to longer queues at intersections, particularly during rush hour when traffic is flowing in a single direction. In response to these issues, cities are introducing dynamic traffic signals that adjust the timing of lights based on real-time traffic conditions.

The road infrastructure is also vulnerable to partial or complete blockages caused by accidents, maintenance, and vehicle mechanical issues, all of which can greatly impact traffic flow. In extreme cases, roads can become gridlocked, with traffic unable to pass through intersections even when they have the right of way. One notable example of this occurred in China[2] in 2010, when drivers experienced traffic congestion that stretched more than 100km over a period of 12 days.

Traffic congestion refers to the overcrowding of certain segments of the road network, which can cause slower-moving vehicles and even complete stoppages. This type of congestion is often periodic and repetitive, occurring at the same times and places, such as during rush hour at bottlenecks and intersections. However, it can also be highly variable and unpredictable, as it can be caused by random events like accidents and maintenance work. Drivers typically rely on static knowledge of the road network to plan their routes without taking into account dynamic traffic conditions and real-time changes to the road

[1] https://thecarconnection.com/news/1122782_americans-spend-18-days-in-cars
[2] https://abcnews.go.com/International/chinas-traffic-jam/story?id=11550037

network. As a result, the chosen route is often not the most efficient one, leading to longer travel times and less optimal use of road traffic. Conventional traffic information systems also often fail to provide accurate, real-time guidance to drivers, as they cannot accurately reflect the dynamic nature of traffic grids.

Fluctuations in traffic demand due to daily variability and special events can also make it difficult to forecast traffic conditions, resulting in unreliable routing solutions. Therefore, any predictive routing solution must consider these fluctuations and try to identify their underlying causes. The effectiveness of a routing solution is typically measured by its ability to avoid bottlenecks or over-capacity conditions, which can be expressed in terms of delay or travel time. Traffic congestion and longer routes contribute to increased vehicle emissions and poorer air quality, which can have negative health effects on drivers, passengers, and even individuals living near congested roads. These emissions can include various harmful pollutants, such as lead, particulate matter, and toxic substances. In addition, the repeated starting and stopping of a vehicle can lead to significant wear and tear, resulting in additional costs for drivers.

1.2 Vehicle safety

Traffic safety features have significantly increased over recent years, providing the newer generation of vehicles with important safety properties. The first generation of safety features included mandatory airbags to reduce injuries during collisions by absorbing the impact of crashes with the vehicle's interior, such as the driving wheel. Lane departure warning systems alert the driver in case of a drifting vehicle from the lane center line. Vehicle over-speed warning devices alert when a driver exceeds the allowed speed limit for a given road segment. Anti-Lock Braking System and Brake Assist system provide better breaking in emergencies. However, these systems still require the driver's intervention to operate and do not take advantage of the recent technological and sensor advancements. Subsequently, newer generations of safety features appeared, such as automatic emergency braking, obstacle detection, and left turn crash avoidance systems which use sensors to detect adjacent vehicles and perform self-activation breaking. Also, driver drowsiness detection systems can detect driver sleepiness levels based on multiple indicators such as steering patterns or even driver face monitoring; these systems are especially helpful for truck drivers who usually take long routes and are susceptible to accidents caused by drowsiness.

However, despite all these safety features, an increasing trend of traffic accident severity emerges. Based on a report by the World Health Organization[3], the number of deaths caused by traffic accidents has reached 1.35 million per year, corresponding to one fatality every 24 seconds. The report also revealed that road accidents are currently the leading cause of death for children and young adults aged between 5 and 29 years. Moreover, a vast disparity appears in the number of accidents between countries, as 93% of fatalities on the roads occur in low- and middle-income countries, despite these countries having approximately only 60% of the world's vehicles; this highlights a non-uniform in progress regarding the reduction of traffic accidents between the regions of the world where the rate of traffic deaths in African countries is the highest with 26.6/100,000 people followed by South-East

[3] https://apps.who.int/iris/handle/10665/277370

Asia 20.7/100,000 people. The loss of life and damage to vehicles and infrastructure can cause countries up to 3% of their Gross Domestic Product (GDP).

According to statistics by the U.S National Highway Traffic Safety Administration[4], driver behavior significantly impacts fatal crashes (Table .1) because speeding is the leading factor for drivers involved in deadly accidents by 17.2%. Driver error also contributes to a large portion of causes of fatal accidents, such as failure to keep in the proper lane, failure to obey traffic signs, and making an improper turn. All these factors can be theoretically avoided and even eliminated using intelligent driving assistant systems.

Table .1: 2019 Fatal Crash Reports for Drivers & Motorcycle Operators.

Behavior	Number	Percent
Driving too fast for conditions or in excess of the posted limit or racing	8,746	17.2%
Under the influence of alcohol, drugs, or medication	5,164	10.1
Failure to yield right of way	3,728	7.3
Failure to keep in proper lane	3,381	6.6
Operating vehicle in a careless manner	3,302	6.5
Distracted (phone, talking, eating, object, etc.)	3,008	5.9
Failure to obey traffic signs, signals, or officer	2,054	4.0
Operating vehicle in erratic, reckless or negligent manner	1,880	3.7
Overcorrecting/oversteering	1,569	3.1
Vision obscured (rain, snow, glare, lights, building, trees, etc.)	1,543	3.0
Drowsy, asleep, fatigued, ill, or blacked out	1,240	2.4
Driving on the wrong way on one-way traffic or the wrong side of the road	1,223	2.4
Swerving or avoiding due to wind, slippery surface, etc.	1,148	2.3
Making improper turn	419	0.8
Other factors	5,549	10.9
None reported	9,196	18.1
Unknown	15,423	30.3
Total drivers (1)	50,930	100.0%

1.3 Understanding traffic flow

1.3.1 Dataset description

To effectively analyze and understand traffic flows, it is important to gather data from a representative sample of roads. The Moroccan center for Road Studies and Research[5] has provided a comprehensive dataset covering a wide range of roadways and traffic conditions from the periods between 2016 and 2017, making it an excellent resource for our study. By utilizing this dataset and applying statistical analysis techniques, we will be able to gain valuable insights into the behavior of vehicles on the road and identify potential areas for improvement in terms of traffic flow and congestion.

The Moroccan center for Road Studies and Research utilize two types of devices to gather detailed and accurate information on traffic flows on selected roadways. These devices are designed to capture a wide range of data about the vehicles passing through the roadway, including their number, type, and speed. This data is then recorded and used to analyze traffic patterns and identify areas of congestion or other issues. By using these

[4] https://cdan.nhtsa.gov/tsftables/tsfar.htm

[5] Le Centre National des Etudes et des Recherches Routières (CNER)

devices, the Center is able to gather comprehensive data on traffic flows and conditions, which can be used to inform the development of solutions to improve traffic flow and reduce congestion.

Data on traffic flows is collected by the center using two types of devices:

- ❑ **Single loop counters:** These devices record the total flow of vehicles across all lanes.
- ❑ **Multi-loop counters:** These devices record the flow of vehicles for each individual lane.

The Moroccan center for Road Studies and Research has put together a dataset containing traffic flow information from roads all over the country. The data is separated into different files, each one containing information about a specific road during a specific period. Along with the raw traffic data, the dataset also includes a classification of vehicles (such as cars or trucks) for each lane, as well as metadata files with additional information like the date and time of data collection, the number of lanes on the road, and the name of the road. The vehicles in the dataset are also classified into five categories based on their length, as seen in Table .2. Examples of the recorded data and the organized dataset can be found in Table .3 and Table .4.

Table .2: Classifying vehicles based on their length.

Class	1	2	3	4
Length	L <5.2m	5.2m <L <6.5m	6.5m <L <10.5m	L >10.5

Table .3: An example of data from a generated file for a two-lane road with vehicle classifications.

Lane 1	Class 1	Class 2	Class 3	Class 4	Lane 2	Class 1	Class 2	Class 3	Class 4
181	85	45	34	17	202	46	63	52	41
389	182	152	42	13	136	46	41	25	24
418	156	124	87	51	118	42	45	19	12
438	205	140	52	41	134	52	43	24	15
316	132	85	67	32	370	196	92	45	37

Table .4: Dataset sample.

Count	Road name	Weekday	Hour
309	RR 413 PK97 MEKNES	Sunday	16.0
790	RN 10 PK 19 Chtouka	Saturday	23.0
20160	RN1 PK222.4 KENITRA	Tuesday	6.0
220	RN 6 PK 71.38 KHEMISSET	Monday	5.0
30	RR401 PK65 KHEMISSET	Friday	4.0
4201	RP3331 PK28 BENSLIMANE	Wednesday	7.0

1.3.2 Data exploration

In this section, we will present a statistical analysis of the traffic flow data contained in the dataset. From Figure .1 and Figure .2, it is clear that there are two major periods of high

traffic volume, occurring between 10 and 17. This pattern is likely due to the increase in commuting traffic during these times, commonly referred to as "rush hour." There is also a noticeable increase in traffic volume on weekends, as shown in Figure .3, which may be due to leisure travel and other non-commuting activities.

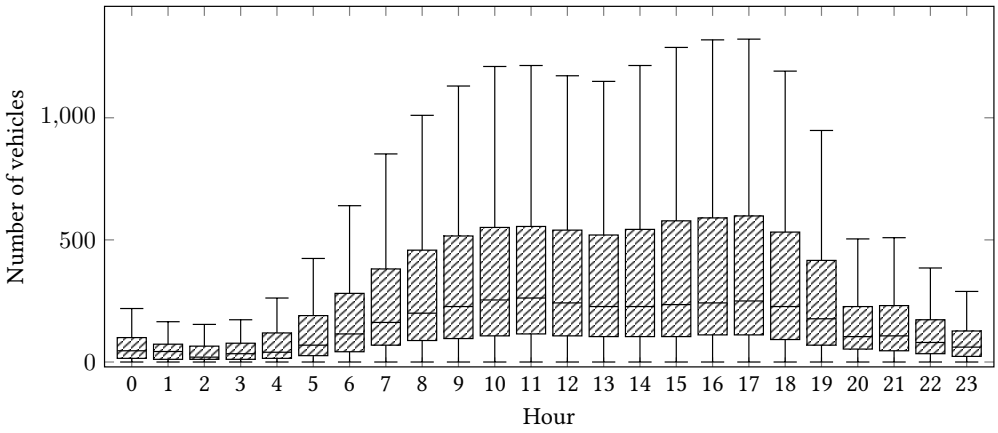


Figure .1: Dataset statistics: grouped by hour.

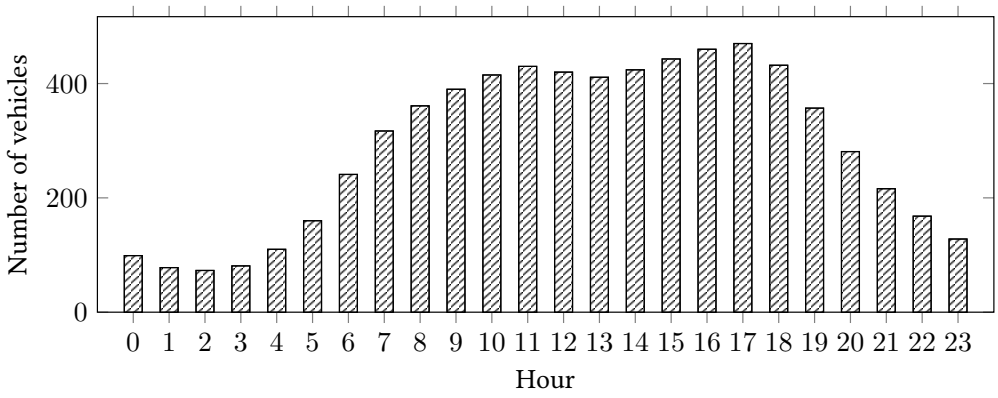


Figure .2: Dataset: grouped by hour.

In conclusion, traffic patterns vary greatly depending on the time of day and day of the week, and this makes it difficult to predict and avoid traffic congestion using traditional methods. One way to address this issue is through the use of advanced traffic control systems for autonomous vehicles. These systems can use real-time data about traffic flow and road conditions to optimize the routing of autonomous vehicles and help reduce congestion. Additionally, autonomous vehicles themselves can help reduce congestion by improving the efficiency of the overall transportation system, as they can communicate with each other and with traffic infrastructure to coordinate their movements and reduce the need for human intervention.

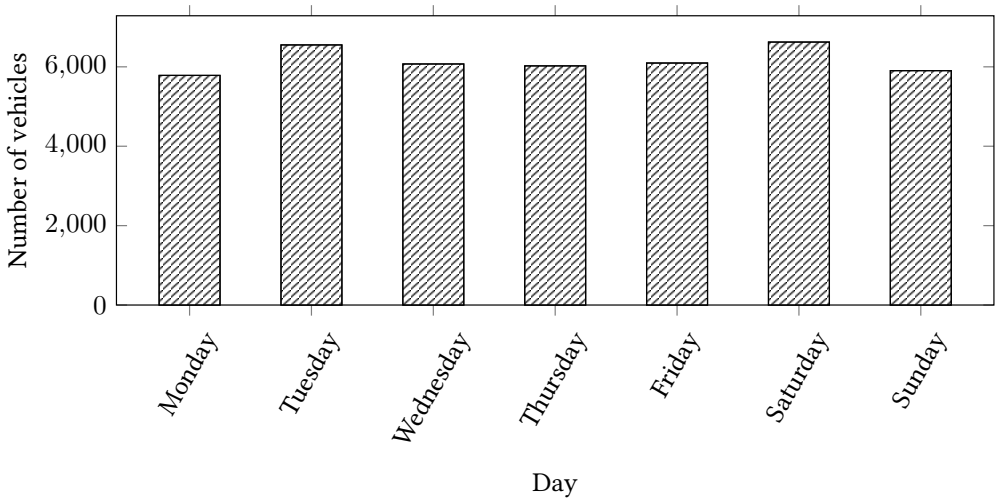


Figure .3: Dataset: grouped by day.

2 Advanced traffic control for autonomous vehicle

2.1 Traffic control

Over the past decade, there have been significant advancements in the field of autonomous vehicles that have enabled researchers to develop safer and more efficient solutions to various traffic-related issues. These advances can be categorized into four main areas:

- Improvements in sensor technology, such as lidars, radar, cameras, 4G, and 5G, have increased performance and reduced costs, making them a viable option for consumer vehicles. These devices provide accurate and comprehensive information about the state of the vehicle and its surroundings and have also become more power efficient, allowing for longer battery life in portable devices.
- Enhancements in the processing capabilities of hardware components, including increased speed for near real-time data processing and improved performance for computationally intensive tasks such as data mining and pattern recognition.
- Access to large amounts of data, including data collected from other vehicles or generated through advanced simulations, which has aided in the training and testing of proposed approaches.
- The availability of high-performance simulation software that allows for the safe testing of new approaches in a virtual environment before implementation in real systems.

The latest technological breakthroughs have enabled the creation and evaluation of control systems for self-driving vehicles that are capable of performing various tasks, including but not limited to, lane tracking, collision identification and prevention, cruise control, enhanced routing, and detection of traffic congestion. These systems often use

vehicle-to-vehicle (V2V) communication to wirelessly transmit information about the vehicle's position, speed, destination, and intentions, as well as the condition of the road, such as the location of congestion and bottlenecks. They can also use vehicle-to-infrastructure (V2I) communication to receive information about road markings, traffic signals, and the state of the traffic grid, including congestion, approaching intersections, construction sites, and parking zones. However, these systems face challenges in terms of standardization and high installation costs.

Overall, the use of advanced artificial intelligence and machine learning techniques, combined with sensory data from cameras, radars, and lidars, has the potential to solve many of the problems related to traffic control and safety in autonomous vehicles. This has led to the development of the field of **Autonomous Vehicles (AV)**.

2.2 Classification of autonomous vehicles

To better understand the level of control a driver has over a vehicle, the SAE International (Society of Automotive Engineers) has developed a classification system [1] for autonomous vehicles that consist of six levels of driving automation, ranging from no automation (level 0) to full automation (level 5). This system provides detailed definitions for each level of automation in the context of vehicles and their operation on roads. The six levels of automation are as follows:

- ❑ **No automation:** The human driver is fully responsible for performing the driving task.
- ❑ **Driver assistance:** The system can assist with either steering or acceleration/deceleration using information about the driving environment, but the human driver is expected to handle all other aspects of the driving task.
- ❑ **Partial automation:** The system can assist with both steering and acceleration/deceleration using information about the driving environment, and it can also monitor the driving environment.
- ❑ **Conditional automation:** The system can handle the full driving task, but it may request human intervention in certain situations and expect an appropriate response.
- ❑ **High automation:** The system may still request human intervention in certain situations, but it no longer assumes that a human response is necessary.
- ❑ **Full automation:** The system is fully responsible for the driving task.

3 Problems formulation

Despite all of the advancements mentioned above, the field of autonomous vehicles still faces many challenges and has areas with rare solutions; that's why in this research, we will propose multiple solutions to enhance the driving experiment and provide safe travel.

One of the main challenges is the need to ensure the safety of passengers and other road users. This requires the development of algorithms and systems that can accurately perceive and respond to their environment in real-time.

Another challenge is the need to improve the overall driving experience for passengers. This includes not only ensuring that the vehicle is able to navigate smoothly and efficiently, but also providing features that enhance the comfort and convenience of the ride.

In this research, we propose to address these challenges by developing multiple solutions that will enhance the driving experience and provide safe travel for autonomous vehicle passengers. Overall, our goal is to contribute to the development of autonomous vehicles that are not only safe but also enjoyable and convenient for passengers to use. By addressing the challenges and areas with rare solutions in the field of autonomous vehicles, we hope to make a meaningful impact on the future of transportation.

The rest of this thesis will be organized as follow:

Chapter 1: will provide an overview of the two main problems that will be addressed in the thesis, namely optimal traffic routing and vehicle overtaking, and highlight and discuss existing research in the literature related to this thesis.

Chapter 2: will focus on the problem of traffic coordination at intersections and propose a decentralized multi-agent architecture that uses deep reinforcement learning to coordinate autonomous vehicles passing through an intersection, with the goal of improving safety, reducing congestion, and providing a more efficient and smooth transportation experience for drivers.

Chapter 3: will focus on the problem of vehicle routing in large cities, and propose a multi-cluster Q-routing model, which partitions the network into smaller clusters and uses Q-routing as the protocol for routing inner and outer traffic for each cluster. By separating inner-cluster traffic from outer-cluster traffic, the model reduces the overall processing power required.

Chapter 4: will focus on the problem of a vehicle overtaking and propose a system that allows vehicles to perform a safe and optimal overtake of under different weather and traffic conditions.

4 List of publications & contributions

All the following publications are co-authored with my thesis director, Prof. Ali Yahyaouy, and my thesis supervisor, Prof. My Abdelouahed SABRI.

4.1 Peer-reviewed Scopus Journals

[1] Lamouik, I., Yahyaouy, A., and Sabri, M. A., "Model predictive control for full autonomous vehicle overtaking," *Transportation Research Record: Journal of the Transportation Research Board*, Jan. 2023.

[2] Lamouik, I., Yahyaouy, A., and Sabri, M. A., "Distributed multi-cluster dynamic q-routing for large size traffic grids," *International Journal of Cloud Computing*, 2022.

4.2 Peer-reviewed International conferences communications

[1] Lamouik, I., Yahyaouy, A., and Sabri, M. A., "Smart multi-agent traffic coordinator for autonomous vehicles at intersections," in *2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, IEEE, May 2017, pp. 1–6.

[2] Lamouik, I., Yahyaouy, A., and Sabri, M. A., "Deep neural network dynamic traffic routing system for vehicles," in *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*, IEEE, Apr. 2018, pp. 1–4.

[3] Jiber, M., Lamouik, I., Yahyaouy, A., and Sabri, M. A., "Traffic flow prediction using neural network," in *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*, IEEE, Apr. 2018, pp. 1–4.

CHAPTER 1

Overview

1 Intoduction

Traffic congestion, vehicle safety, and advanced traffic control have become critical areas of research and development in the field of autonomous vehicles. As cities grow more populated, traffic congestion poses challenges such as wasted time, increased pollution, and compromised quality of life. The safety of passengers and pedestrians remains a top priority for the widespread adoption of autonomous vehicles. Researchers are actively exploring innovative solutions to address these challenges and revolutionize transportation networks. They leverage cutting-edge technologies and intelligent algorithms to promote efficient traffic flow, reduce travel times, and enhance overall road safety.

Research in this field encompasses various interconnected aspects. Intelligent traffic management systems, predictive modeling, and data analytics are being utilized to address traffic congestion. Advanced algorithms optimize traffic signal timings, dynamically reroute vehicles based on real-time conditions, and implement adaptive traffic control strategies. By utilizing data from connected vehicles and infrastructure sensors, these systems respond to changing traffic patterns and congestion hotspots, resulting in improved traffic flow.

Ensuring vehicle safety is of paramount concern. Autonomous vehicles are equipped with advanced sensors such as LiDAR, cameras, and radar, enabling them to perceive their surroundings and make informed decisions. Researchers continuously enhance sensor capabilities and develop sophisticated algorithms for better object detection, recognition, and tracking. The goal is to minimize the risk of collisions and pedestrian accidents by enabling autonomous vehicles to accurately interpret their environment and take appropriate real-time actions.

Advanced traffic control systems and autonomous vehicles are considered potential solutions for enhancing safety. By eliminating human error, autonomous vehicles consistently follow traffic rules, maintain safe distances, and react swiftly to hazards. Integration of advanced traffic control systems allows seamless coordination and communication among vehicles, optimizing traffic flow and reducing the risk of collisions. This synergy holds great promise in significantly reducing accidents and saving lives on the roads.

2 The problem of optimal traffic routing

2.1 Persistent Traffic Congestion and Induced Demand

Despite efforts to address traffic congestion in modern cities through the construction of bigger road infrastructure such as larger highways and tunnels by spending billions of dollars each year[1], the problem persists. This is due in part to the phenomenon of induced demand[7], which states that any increase in road capacity will eventually be filled by more traffic. Additionally, as traffic demand continues to grow, it will always exceed the capacity of the traffic network. Many cities have already reached the point where the traffic flow is greater than the capacity of any additional lanes. These efforts to address traffic congestion often come at a high cost, with governments and city planners spending billions of dollars each year on such projects.

Furthermore, the expansion of modern cities and the increased availability of vehicles have contributed to traffic congestion, as millions of people spend hours each day commuting to work. This not only results in wasted time and decreased productivity, but has also been linked to various health problems [8]. Physical symptoms such as headaches, backaches, and digestive issues have been reported, as well as mental disorders such as sleep disturbances, fatigue, and concentration problems. These longer commutes can have serious consequences for both individuals and society as a whole.

2.2 Delivery Services and Last Mile Problem

Moreover, the increasing prevalence of delivery services, focused on cost minimization, has necessitated the use of efficient traffic management and routing systems. These systems must effectively regulate traffic to minimize travel time by avoiding congested roads and long red lights. Additionally, the rise in online shopping and home delivery has also led to a demand for optimal routing algorithms, aimed at reducing expenses and promptly serving the maximum number of customers.

The final leg of a product's delivery journey, known as the "last mile," is typically the most critical to ensuring customer satisfaction. This stage involves transporting the item from an overseas warehouse to an airplane or shipping container, then to the back of a truck, and ultimately to the customer's doorstep. Despite its importance, the last mile is also the most expensive and time-consuming aspect of the shipping process. In fact, it is often referred to as the "last mile problem" because it can account for as much as 28% of the total delivery cost.

A new emerging trend in the field of transportation in recent years is Mobility-as-a-Service (MaaS), which aims to offer a complete solution for all transport needs (vehicle rental, logistics services, and personal travel). This trend is driven by various innovations, such as carpool and ride-share companies and bicycle-sharing systems. These services rely heavily on minimizing travel time to save on the cost of fuel and also maximize the number of trips. Therefore, they can naturally benefit from better routing solutions.

[1] https://www.moroccoworldnews.com/2020/12/328182/morocco-to-enlarge-highways

2.3 Environmental and Economic Considerations

Statistics show that the average time spent on each trip is reduced with efficient driving strategies, improving the efficiency of the overall traffic flows and reducing fuel consumption and pollution. In fact, in 2017, road transport contributed 21% of the EU's total carbon dioxide (CO2) emissions, the main greenhouse gas. Therefore, the environmental cost of the higher emissions (CO2, CO, NOx, particulate matter, and others) is a significant burden on the effort to reduce net greenhouse gas emissions. Fuel consumption can also benefit from better routing policies since vehicles cannot move at their optimal driving speed as traffic flow decreases. Also, at peak congestion, vehicles spend considerable time idling in traffic, leading to the degradation of the vehicle caused by repeated acceleration, deceleration, gear changes, and brake applications. All of the factors above lead to higher Vehicle Operating Costs (VOC) [9]. Consequently, effective traffic routing is crucial to solving the congestion problem. Essentially, it optimizes resource utilization and minimizes average trip length, travel time, and vehicle fuel consumption. Nonetheless, this task has become increasingly difficult for traditional static algorithms as the size of the network increases, especially since traffic networks are inherently highly dynamic and change frequently. Therefore, more research is needed to design better path planning and routing systems that can adapt well to these circumstances.

2.4 Limitations of Traditional Routing Approaches

In the realm of connected and automated vehicles, traffic planning and routing are significant research areas. As the volume of vehicles on the roads continues to rise, it is becoming progressively more difficult to identify optimal routes that are both fast and safe. The state and action spaces have also grown in size, making it challenging to find solutions within a reasonable time frame using traditional methods. To tackle these challenges, researchers have suggested employing machine learning algorithms, including reinforcement learning, to model traffic patterns and anticipate the best routes based on historical data. Another strategy involves facilitating communication between vehicles and infrastructure, such as traffic lights and road signs, to exchange information and coordinate vehicle movements in real-time. Addressing the challenges associated with the increasing number of vehicles on the roads will necessitate the development of new technologies and approaches to traffic planning and routing, which are intricate problems.

Consequently, optimizing routing is essential for improving the efficiency of the traffic network as it reduces travel time for vehicles. However, finding the best routing solution in a large traffic grid is challenging due to computational and memory constraints. As a result, most routing approaches aim to minimize the average cost of all paths for each vehicle without considering the impact on other cars and the overall performance of the grid. This can be a significant issue as it fails to take into account the effect of each path on the rest of the traffic network.

2.5 Routing methods classifications

2.5.1 Adaptability classifications

Routing algorithms can be classified based on their ability to adapt to changes in the traffic grid, as either dynamic (adaptive) or static (non-adaptive). These algorithms can also

be classified based on the level of determinism in their approach, as either deterministic or stochastic. Additionally, routing algorithms can be categorized as either reactive or predictive, depending on whether they respond to changes in real-time or try to anticipate future changes. Finally, routing algorithms can be centralized, meaning they rely on a central authority to make routing decisions, or decentralized, meaning that routing decisions are made locally.

1. **Static Vs. Dynamic:** Static routing is the most basic form of a vehicle routing system. It relies on predetermined configurations of routing paths that are not affected by changes in the network state, such as distance, number of intersections, etc. In static routing, each destination has only one entry in a routing table, and the route is fixed after it is assigned. This means that static routing cannot adapt to changes in the traffic network, such as road closures, accidents, maintenance, or fluctuations in traffic during peak hours. However, static routing has the advantage of faster path computation compared to dynamic routing. Algorithms like A* and Dijkstra's are commonly used for static routing, but as the number of nodes in the network increases, these algorithms tend to have very high time complexity.

   On the other hand, dynamic routing uses the real-time state of the network to continuously evaluate and optimize the route based on changes in the network's topology or congestion. The advantage of dynamic routing is its ability to adapt to frequent changes in traffic flow without requiring intervention from the system administrator. Many dynamic routing algorithms exist, such as dynamic shortest path (DSP).

   While static routing is well-suited for real-time applications because of its fast calculation times and scalability, it may not be effective if the network topology changes frequently.

2. **Centralized Vs. Decentralized:** In theory, if a centralized routing system has complete and up-to-date information about the state of the traffic grid, as well as sufficient memory and computation power it can provide optimal routing for each vehicle in the grid. However, this type of system can be expensive to implement and maintain, especially in high-traffic networks. As a result, decentralized traffic routing algorithms have been developed, which divide the network into smaller sub-networks (clusters) and allow for more practical approaches.

   In a decentralized routing system, each node only keeps track of its local traffic, reducing the amount of computation and memory usage significantly. Ideally, an optimal routing system should provide decentralized solutions that do not rely on online centralized infrastructure and only use sensors and communication with other machines. Decentralized routing offers many benefits, such as the reduced risk of complete system failure due to distributed nodes that can offer backups, and increased immunity against attacks. However, centralized systems may be more attractive due to their simpler maintenance requirements.

3. **Reactive Vs. Proactive:** Route planning can also be classified as reactive algorithms, also known as on-demand routing, which means that a route is determined based on current traveling conditions only when needed. These algorithms do not maintain a routing table or any other information about the network in advance. Instead, they

use a search mechanism, such as flooding or breadth-first search, to find a route from the source to the destination when a packet needs to be transmitted.

Generally, reactive routing tends to be more efficient in dynamic networks than proactive routing, but it may have long waiting times for route calculation. Some popular reactive algorithms include Ad-hoc On-demand Distance Vector (AODV) and Dynamic Source Routing (DSR).

Examples of reactive routing algorithms include flooding, breadth-first search, and depth-first search. Flooding is a simple reactive routing algorithm in which a packet is broadcast to all nodes in the network, ensuring that it will reach the destination, but it can also generate a lot of unnecessary traffic and may not be very efficient. Breadth-first search is a more efficient reactive routing algorithm in which the packet is sent to all neighboring nodes first before being sent to the next level of nodes, guaranteeing that the shortest path to the destination will be found. However, it can still result in a lot of unnecessary traffic if the destination is not close. Depth-first search is a reactive routing algorithm in which the packet is sent along the first path encountered until the destination is reached or the search reaches a dead end. This algorithm may be faster than a breadth-first search, but it may not always find the shortest path to the destination.

On the other hand, predictive algorithms, also called proactive algorithms, are table-driven routing algorithms in which each node in the network stores and maintains a complete routing table in advance. These algorithms periodically exchange routing information with their neighbors to update the routing table and determine the best routes to all destinations in the network. When a packet needs to be transmitted, the source node can look up the destination in the routing table and use the pre-computed route to send the packet.

Reactive routing algorithms are typically more responsive to changes in the network, but they may be slower and less efficient than proactive algorithms. Proactive routing algorithms are generally faster and more efficient, but they may not be as responsive to changes in the network. An example of a proactive routing algorithm is Destination-Sequenced Distance Vector routing (DSDV), which uses current load conditions and historical data and statistical techniques to estimate the future state of the network.

Examples of proactive routing algorithms include the link-state routing algorithm and the distance-vector routing algorithm. The link-state routing algorithm is a proactive routing algorithm in which each node maintains a complete map of the network and exchanges information with other nodes to update their maps. They use this information to compute the best routes to all destinations in the network, ensuring that the shortest path is used. However, this algorithm can be computationally expensive and may not be suitable for very large networks. The distance-vector routing algorithm is a proactive routing algorithm in which each node maintains a table of the distances to all other nodes in the network and exchanges information with its neighbors to update the table. They use this information to determine the best routes to all destinations in the network, but it may not always find the shortest path.

4. Deterministic Vs. Stochastic: Deterministic routing algorithms are those that consis-

tently produce the same output for a specific input, with no randomness involved. This means that the output of a deterministic algorithm is entirely determined by its input and the algorithm itself. Although deterministic algorithms are often simple to understand and analyze, they may not always be the most effective or efficient way to solve a problem. Some examples of deterministic routing algorithms include shortest path algorithms like Dijkstra's algorithm and the Bellman-Ford algorithm and greedy algorithms that make decisions based on locally optimal choices.

Shortest path algorithms are used to find the shortest or most efficient route between two points in a network. These algorithms typically use a distance or cost metric to determine the best route, and they always produce the same result for a given input. Greedy algorithms are a type of deterministic algorithm that make decisions by choosing the locally optimal option at each step. For example, a greedy routing algorithm might always choose the next hop that minimizes the total distance to the destination, without considering the long-term effects of its decisions.

Another example of a deterministic routing algorithm is a fixed routing algorithm, in which each node in the network has a predetermined route to every other node. This type of algorithm is simple to implement, but it is not very flexible and may not always find the most efficient route.

On the other hand, stochastic routing methods involve a small amount of randomness. Decisions made by these algorithms often depend on probabilistic data, such as the likelihood that a particular route will be effective. While stochastic algorithms are often more versatile and flexible than deterministic algorithms, they may be more difficult to comprehend and analyze.

Stochastic routing algorithms encompass a range of techniques, including genetic algorithms, simulated annealing algorithms, and ant colony optimization algorithms. Genetic algorithms use evolutionary principles such as reproduction, mutation, and selection to discover solutions to optimization problems in a stochastic manner. Similarly, simulated annealing algorithms employ randomness to explore the solution space and prevent getting trapped in local minima or suboptimal solutions. In contrast, ant colony optimization algorithms utilize stigmergy, which involves indirect communication through the environment, to find solutions to optimization problems in a stochastic manner.

Stochastic routing algorithms are often used in complex networks where it is difficult to predict the best route in advance. These algorithms are typically more flexible and adaptable than deterministic algorithms, but they may be more difficult to analyze and understand.

5. Periodic Vs. Event-based: In traffic routing, periodic updates refer to updates that are scheduled to occur at regular intervals, such as every 30 seconds or every 5 minutes. These updates allow routers to exchange routing information and maintain accurate and up-to-date routing tables. On the other hand, event-driven updates are triggered by specific events or actions, such as when a link goes down or a new link is established. These updates enable routers to quickly adapt to changes in the network and maintain accurate routing information.

Hybrid updates are a combination of periodic and event-driven updates. In a hybrid update system, updates may occur at regular intervals and also be triggered by

specific events or actions. Hybrid updates can offer the benefits of both periodic and event-driven updates, allowing a system to maintain a consistent level of information while also being able to respond to changing conditions in real-time.

All of these update types can be used to exchange routing information between routers in a network. Periodic updates can be useful for maintaining a consistent level of routing information, but they can also result in unnecessary traffic if the network is not changing frequently. Event-driven updates can reduce unnecessary traffic, but they may not be as effective at maintaining a consistent level of routing information. Hybrid updates can provide the benefits of both, allowing routers to adapt to changes in the network while also maintaining a consistent level of routing information. The choice between the different update types will depend on the needs of the network and the type of routing protocols being used.

2.5.2 Problem based classification

Alternatively routing algorithms may be classified based on the type of problem being addressed. Here are a few examples of different types of vehicle routing algorithms:

1. Single-vehicle routing algorithms These algorithms are designed to find the optimal route for a single vehicle to visit a set of locations. They may be used, for example, to optimize the delivery route for a single delivery truck.

- ❑ **Traveling salesman problem (TSP):** One of the most well-known and classical single-vehicle routing algorithms is the TSP [10], which involves finding the shortest possible route for a salesman to visit a set of cities, starting and ending at a designated home city. Several algorithms can be used to solve the TSP, including exact methods such as branch and bound and heuristic approaches like the nearest neighbor algorithm and the 2-opt algorithm.

  Recently, a new algorithm called NeuroLKH has been developed [11] that combines deep learning with the traditional heuristic Lin-Kernighan-Helsgaun (LKH) to solve the TSP. NeuroLKH uses a Sparse Graph Network (SGN) to generate edge scores and node penalties, which guide the searching process of LKH. Another single-vehicle routing algorithm is the two-phase algorithm introduced in [12], which is based on an elementary shortest path algorithm with resource constraints. In the first phase of this algorithm, all non-dominated feasible routes are generated, and in the second phase, some of these routes are selected and sequenced to form the vehicle's workday. A recent variant of the TSP, known as the traveling salesman problem with a drone station (TSP-DS), has gained significant attention in the literature, with studies such as [13], [14], and [15] investigating its properties. In the TSP-DS, a drone station is used as a storage facility for drones and charging devices, usually located far from the package distribution center.

- ❑ **Vehicle routing problem with time windows (VRPTW):** VRPTW is a variant of the TSP that involves delivering goods to customers within specific time windows [16], [17]. The goal of the VRPTW is to find the optimal route that minimizes the total travel time, subject to the time window constraints. In [18] The authors propose a fuzzy optimization model for the FEVRPTW based on

credibility theory, which allows for partial recharging and develops an adaptive large neighborhood search (ALNS) algorithm enhanced with the fuzzy simulation method to solve it. In [19], a solution approach using a tabu search heuristic that considers different shortest paths between customers at different times of the day is proposed to solve the VRPTW. Another solution approach for the VRPTW is proposed in [20], where the authors use a simulated annealing (SA) algorithm to minimize the total cost, including traversing cost, vehicle employment cost, and exit penalty from permissible time windows.

- ❑ **Pickup and delivery problem (PDP):** (PDP) is a variant of the VRPTW that involves not only visiting a set of customer locations but also picking up and delivering packages at those locations [21]. The goal of the PDP is to find the optimal route that minimizes the total travel time, subject to the time window constraints and the package pickup and delivery constraints. In [22], the authors study the concept of crowdsourced delivery, which involves using excess capacity on journeys that are already taking place to deliver packages, as a way to improve the efficiency and reduce the costs of last-mile delivery for Internet retailers. In [23], the authors study the stochastic dynamic pickup and delivery problem, which involves a fleet of drivers delivering food from restaurants to order customers. They propose an anticipatory customer assignment (ACA) policy to dynamically control the fleet and avoid delays for customer deadlines by introducing a time buffer to reduce the likelihood of delays. In [24], the authors propose two meta-heuristics, a Large Neighbourhood Search (LNS) and a Genetic Algorithm (GA), for solving the pickup and delivery problem with transshipment (PDPT). The PDPT involves finding routes that satisfy requests between pickup and delivery nodes and allows for requests to be transferred from one vehicle to another at transshipment nodes. The authors conclude that the GA outperforms the LNS and the existing solutions in the literature. On the other hand, metaheuristic solutions have also been investigated for the PDP in [25] and [26].
- ❑ **Vehicle routing problem with backhauls (VRPB):** The VRPB is a variant of the VRPTW in which the vehicle not only has to visit a set of customer locations but also has to return to a central depot with a load of goods. The goal is to find the optimal route that minimizes the total travel time, subject to the time window constraints and the backhaul constraints. The authors in [27] propose a general-purpose formulation based on the open vehicle routing problem that is able to eliminate the possibility of generating solutions with sub-tours, while still allowing for Hamiltonian paths and tie-arcs to connect them. In [28] A heuristic multi-start biased-randomized is proposed as a solution, using an iterative method to solve smaller instances of the homogenous-fleet version of the problem and using the subsolutions as partial solutions for the original heterogeneous instance. In [29] the authors proposed a solution based on integer linear programming formulation and a hybrid differential evolution algorithm involving a genetic operator and a fuzzy logic controller. with the objective of minimizing total cost.
- ❑ **Rural postman problem (RPP):** The RPP [30] is a variant of the TSP in which the vehicle must visit a set of customer locations that are not necessarily connected by roads. The goal is to find the optimal route that minimizes the total

travel time, subject to connectivity constraints. In [31] the authors present optimality conditions, a worst-case analysis, and a compact integer linear programming formulation for the problem of RPP.

2 .**Multi-vehicle routing algorithms** Multi-vehicle routing algorithms are designed to calculate the optimal routes for a fleet of vehicles to visit a set of locations, such as delivery trucks. In [32], a mixed integer programming paradigm was proposed to model the delivery routing optimization problem (DROP) for multiple simultaneously-operating vehicles, taking into account their energy constraints. The problem was then solved using a multi-chromosome genetic algorithm, which also optimized the number of delivery vehicles. [33] introduced a novel reinforcement learning algorithm called the Multi-Agent Attention Model for solving the multi-vehicle routing problem with soft time windows (MVRPSTW) in urban logistics distribution systems. This algorithm used an encoder-decoder framework with attention layers and a multi-agent reinforcement learning method with an unsupervised auxiliary network. In [34], AND/OR precedence constraints were added to the multi-vehicle routing problem, involving mandatory or optional dependencies between nodes in the routing process. A decomposition-based technique using an enhanced version of Benders' decomposition, called the branch and search algorithm, was proposed to solve the problem. Another variant of the multi-vehicle routing problem is the multi-vehicle routing problem with dense, dynamic obstacles (MVRP-DDO), which takes into account nonholonomic constraints and the presence of dense, dynamic obstacles. [35] proposed a multi-abstraction search approach to compute an executable plan for the drilling operations of multiple machines in a constrained environment, which was evaluated in terms of makespan and computation time with a focus on meeting the hard industrial requirements of these metrics. Finally, in [36], a mathematical model and genetic algorithm were presented for the dynamic multi-vehicle routing problem, in which a fleet of vehicles delivers goods to a set of demand points and can transfer goods between each other at demand points to better satisfy late demands and reduce costs.

3. **Capacitated vehicle routing algorithms** Capacitated vehicle routing algorithms are designed to find the optimal routes for a fleet of vehicles, subject to constraints on the capacity of each vehicle. They may be used, for example, to optimize the routes for a fleet of delivery trucks that have limited cargo capacity.

   In [37] the authors proposed a hybrid algorithm that combines the firefly algorithm (FA) with two types of local search and genetic operators in order to solve the capacitated vehicle routing problem (CVRP). Multiple FA populations are used, and they communicate and exchange solutions in order to maintain diversity and prevent the algorithm from falling into local optima. Another approach based on the firefly algorithm (FA) called CVRP-FA was proposed in [38]to solve the CVRP by combining FA with two types of local search and genetic operators to improve the quality of the solutions and accelerate convergence. Heuristic algorithms are also used for solving the capacitated vehicle routing problem in [39], which combines tabu search and adaptive large neighborhood search. In [40] a multi-start algorithm for solving the capacitated vehicle routing problem with 2D loading constraints (2L-CVRP), which allows for the rotation of goods. The algorithm is based on metaheuristics such as the Constructive Genetic Algorithm (CGA), Tabu Search (TS), and Large Neighborhood

Search (LNS). An improved flower pollination algorithm (IFPA) was proposed in [41] to solve the CVRP. The IFPA was designed to enhance the global search capability of the traditional flower pollination algorithm (FPA) by introducing a random jump perturbation in the global pollination phase and updating the switching probability according to the global optimal value of each iteration.

4. **Time-constrained vehicle routing algorithms** These algorithms are designed to find the optimal routes for a fleet of vehicles subject to time constraints, such as delivery deadlines or time windows during which deliveries must be made.

   In [42], a path planning algorithm for a multi-AGV system in a warehouse environment was proposed, based on the A* heuristic algorithm and taking into account probabilistic time constraints and conflict avoidance strategies. The effectiveness of the algorithm was demonstrated through numerical simulation experiments in a warehouse setting. In [43], the authors proposed two algorithms based on mixed-integer programming models to solve the time-constrained maximal covering routing problem, which involves finding the optimal routing for a fleet of vehicles. In [44], a method for recommending paths to drivers that takes into account their historical GPS data and personal preferences was proposed. This method involves extracting critical intersections from the historical data, determining time-varying paths between these intersections, and using an improved dynamic A* search algorithm on a simplified road network to find candidate paths that meet time constraints. In [45], a mixed integer programming formulation was developed to take into account the UAV's turning radius, and a simulated annealing heuristic was also developed to solve larger problems that may not be feasible with the MIP approach.

2.6 Routing algorithms

There are numerous routing algorithms in the literature, each with its advantages, disadvantages, and enhancements. In this section, we will give a brief overview of some of the more commonly used routing algorithms.

1. **Link state routing (LSR):** LSR is a proactive routing protocol that maintains a complete map of the network architecture in each router. It is table-driven and utilizes the Shortest Path First (SPF) algorithm, also known as Dijkstra's algorithm, to determine the best route to a destination. In this approach, each router sends a Link-State Advertisement (LSA) to other routers to update them on the status of its directly connected links whenever there is a change in the network. As a result, every router maintains a map of the entire network. One example of a link-state protocol is OSPF (Open Shortest Path First).

   Link state routing protocols tend to converge more quickly and are less prone to routing loops compared to distance vector routing protocols. However, they require more CPU and memory resources, which can make their implementation and support more difficult and costly. In addition, the hierarchical structure used by link-state protocols limits the distance that a Link-State Advertisement can reach. To address this issue, the Multi-Channel Link-State Routing protocol (MCLSR) [46] was proposed for use in multichannel mesh networks to minimize broadcast overhead due to link-state propagation. Another solution is the dynamic multipath algorithm proposed in

[47], which computes multiple next-hops for each destination in a link-state network to ensure a loop-free path.

Overall, link-state routing has the advantage of faster convergence and reduced risk of routing loops, but it can be resource-intensive and may not be suitable for large networks due to the flooding of the network with Link-State Advertisements.

2. **Optimized Link State Routing Protocol (OLSR):** OLSR is a proactive link-state routing protocol that uses hello and topology control (TC) messages to discover and disseminate link-state information throughout an ad hoc network. This allows each node to determine the shortest hop-forwarding paths to all other nodes in the network and establish next-hop destinations. OLSR uses a link-state method to proactively find the most effective path between nodes and employs the dynamic Multi-Point Relay (MPR) approach, in which only a selection of surrounding nodes act as relays to transmit data, allowing every node to communicate with an MPR within a single hop. MPRs exchange localized network information to maintain network-wide routing pathways, which enables every MPR to have a full routing table while also reducing the volume of topology-control messages.

   Several variations of OLSR have been proposed to address its limitations, such as high node discovery time and resource requirements in large networks. For example, in [48], a fuzzy logic-based routing approach called OLSR+ was proposed to improve the use of OLSR in FANETs by improving the estimation of link lifetime and using a fuzzy mechanism to select MPR nodes. In [49] and [50], an optimized OLSR was proposed that uses multi-beam transmissions and directional antennas to achieve multi-path concurrent data delivery with extended signal range. Other proposed variants include the use of a Multipath Dijkstra Algorithm to offer multiple paths for sparse and dense topologies in [51], the combination of quantum genetic strategy and Q-Learning to adapt to dynamic network topology changes in [52], and the use of metaheuristics such as the Bat Algorithm (BA) in [53] and Ant Colony in [54] to optimize data transmission.

3. **Distance Vector Routing (DVR):** DVR[55], known also as the Bellman-Ford routing algorithm and the Ford-Fulkerson algorithm is a method in which each router maintains a Distance Vector table containing the distance between itself and all possible destination nodes in the network. In this approach, distance is calculated based on the number of routers that a packet must travel through, with each router counting as one hop. Some distance vector protocols also consider additional variables such as network latency that can affect traffic on a particular path.

   In distance vector routing, each router sends all or part of its routing table to its neighbors regularly, and the neighbors update their routing tables to reflect any changes. This process, known as routing by rumor, allows the network to stay updated on the current state of the network. However, it can also lead to the count-to-infinity problem, in which two routers continue to increase the distance between them indefinitely due to incorrect updates. In addition, distance vector routing tends to have slower convergence and requires large routing tables as every router must be aware of every other router in the network.

   To address these issues, various variants of distance vector routing have been proposed. For example, in [56], the authors propose using one-way hash functions to

protect the network against Denial-of-Service (DoS) attacks. In [57], the authors propose the Binary State Distance Vector Routing (BSDVR) to provide routing in a partitioned network. And in [58], the authors propose a solution to packet duplication using pseudo diameter.

4. **Destination-Sequenced Distance-Vector Routing (DSDV):** DSDV [59] was developed to address the routing loop problem in Distance Vector Routing (DVR). In DSDV, each routing table entry is associated with a sequence number that helps to identify stale entries. However, this approach requires a new sequence number to be assigned with every network change, which makes it less suitable for highly dynamic or large-scale networks.

   To improve the performance of DSDV in Mobile Ad-Hoc Networks (MANETs), which are characterized by their dynamic nature, several extensions have been proposed. For example, in [60] and [61], the authors propose extensions to DSDV for use in MANETs. In [62], the authors propose a clustering approach to improve network throughput and use the Ad-Hoc On-Demand Distance Vector (AODV) routing protocol for inter-cluster communication. Overall, DSDV offers a solution to the routing loop problem in DVR, but it may not be well-suited for highly dynamic networks.

5. **Ad-Hoc On-Demand Distance Vector (AODV):** AODV [63] is a mobile and wireless network routing protocol. The protocol supports both unicast and multicast routing and constructs routes to destinations on demand. Because routes between nodes are only generated following a request from a source node, (OADV) takes an on-demand approach to route discovery. This method has the advantage of reducing the amount of traffic communication between nodes. The destination sequence number is used by (AODV) to calculate the fastest path to the destination. Only nodes with a smaller hop count update their path information if the current packet's destination sequence number is greater than or equal to the previous packet's destination sequence number.

   Multiple attempts to improve (AODV) have been proposed. In [64] To reduce routing overhead and memory consumption, 'LOADng' does not maintain a routing table for all destinations in the network, but initiates route discovery only when there is data to be sent there. In [65] proposed a protocol based on the multicast operation of the AODV routing protocol, a Reliable and Energy-Aware Multicast Ad-hoc On-demand Distance Vector (REA-MAODV) routing protocol is developed to reduce the wastage of resources.

   In [66] the authors proposed an event-driven routing protocol based on Multicast Ad hoc On-demand Distance Vector (ECA-MAODV) using a fuzzy set to create a multicast tree. and in [67] a hybrid network with a delay tolerant network (DTN) is proposed to detect black hole attack against the network and increases the throughput ratio and packet transmission ratio and decreases the overall delay.

6. **Global State Routing (GSR):** GSR [68] is a proactive routing scheme that is based on the principles of Link State Routing (LSR) and aims to improve upon it by avoiding the flooding of routing messages. In GSR, each node maintains a Neighbor list of its neighbors, a Topology table containing link state information reported by the destination node and the timestamp of this information, a Next Hop table containing the next hop to which packets for a particular destination should be forwarded, and

a Distance table containing the shortest distance to each destination node. However, GSR has the drawback of large update messages, which results in high bandwidth consumption and operational cost.

Enhancements to GSR have been proposed to address these limitations. For example, in [69], a GSR enhancement was proposed that includes protection against denial of service attacks (DoS) by including packet priority and enhancing node discovery. In [70], GSR was chosen as the routing algorithm for the analysis of route lifetime.

7. Fisheye State Routing (FSR): FSR[71] is a type of hierarchical routing protocol that is designed to improve the performance and scalability of routing in large and complex networks. It is an improvement upon Global State Routing (GSR) and aims to reduce the size of update messages, which can be large and consume high bandwidth in GSR. In FSR, each update message does not contain information about all nodes. Instead, it exchanges information about closer nodes more frequently, using the Fisheye scope to define the set of nodes that can be reached in a certain number of hops. This reduces the size of the message by propagating entries in the routing table corresponding to nodes within the smallest scope to neighbors with the highest frequency.

The FSR protocol divides the network into a hierarchy of domains, with each domain being composed of a number of nodes. The nodes in each domain maintain routing information only for the nodes within their domain and the nodes in the adjacent domains. This helps to reduce the amount of routing information that each node needs to maintain, making the routing process more efficient and scalable. FSR is effective in improving the performance and scalability of routing in large and complex networks and can adapt to changes in the network over time.

In [72], the use of FSR in Flying Ad-hoc Networks with limited energy and bandwidth resources was investigated, and it was concluded that FSR outperformed AODV, DSDV, DSR, OLSR, and TORA. In [73], FSR was combined with Q-learning to reduce FSR computations and eliminate node dependencies. However, in [74], it was concluded that DSR outperformed FSR in VANET in terms of throughput and end-to-end delays.

8. Clusterhead Gateway Switch Routing (CGSR): CGSR[75] is a table-driven routing protocol that is based on the principles of Destination-Sequenced Distance-Vector (DSDV) routing. In CGSR, nodes are grouped into clusters and each cluster is assigned a cluster head, which is computed using a distributed clustering algorithm. Each node maintains two tables: a cluster-member table that lists the cluster head of all nodes in the network, and a routing table that determines the next hop to reach the destination cluster.

Several enhancements to CGSR have been proposed to improve its performance. In [76], an Enhanced CGSR (ECGSR) was proposed that is based on the Ad Hoc On-Demand Distance Vector (AODV) routing protocol. In ECGSR, the cluster head performs congestion monitoring by reducing the number of transmissions and creating routes on demand. In [77], fuzzy rules are combined with a particle swarm optimization (PSO) based CGSR algorithm to produce an energy-efficient routing system. In [78], malicious and misbehaving nodes in a CGSR-based network are identified.

9. Wireless Routing Protocol (WRP): WRP[79] is a distance-vector routing protocol that uses tables to manage routing information. Each node in the network maintains four main tables: a Distance table, a Routing table, a Link-Cost table, and a Message Retransmission list.

The Distance table contains information about the destination, next hop, distance, and predecessors for each destination and neighbor. The Routing table includes the destination address, next hop, distance, predecessor, and a marker for each destination, which indicates whether the entry corresponds to a simple path.

The Link-Cost table provides the link cost to each neighbor, as well as the number of periodic update periods that have passed since the node received an error-free message from that neighbor. The Message Retransmission list records which updates in an update message need to be retransmitted and which neighbors need to acknowledge the retransmission. It also includes the sequence number of the update message, a retransmission counter, and a list of updates sent in the update message.

10. Dynamic Source Routing (DSR): Dynamic Source Routing (DSR) [80] is a reactive, on-demand routing protocol designed specifically for multi-hop wireless ad hoc networks of mobile nodes. It is self-organized, self-configured, and capable of performing route discovery without the need for any existing infrastructure. The "Route Discovery Mechanism" in DSR is triggered when a sender attempts to send a packet to a destination with an unknown route and allows the sender to select and control the routes used for their packets, supporting the use of multiple routes to any destination and ensuring loop-free routing. When multiple routes to a destination are found, the route with the minimum hop count is selected. DSR also uses the "Route Maintenance" mechanism to detect changes in the network topology, such as broken routes, and allows the sender to use any other known routes to the destination or invoke Route Discovery again to find a new route for the remaining packets. DSR does not require periodic routing advertisements, link status sensing, or neighbor detection packets of any kind, making it more scalable.

11. Temporally Ordered Routing Algorithm(TORA): TORA is an on-demand, adaptive routing protocol for multi-hop networks. TORA is based on a link reversal algorithm and attempts to offer loop-free and scalable routing. TORA's objective is to minimize the communication overhead associated with adapting to network topological changes by adapting local topological changes in ad hoc networks. TORA has the benefit of supporting multiple routes to transmit data packets between source and destination. TORA uses the concept of the router's "height" to direct the packet "downstream" by creating a directed acyclic graph (DAG) which is rooted at the destination. If some routers lose their last downstream link, some directed paths may temporarily cease to lead to the destination. When router heights are re-selected, there will be a sequence of reversals of direct links, which will re-orient the routing structure so that all directed paths are once again redirected to the destination.

12. Signal stability-based adaptive routing (SSA): SSA is a type of routing algorithm that uses signal stability as a metric to determine the best routes in a wireless network.

In this type of algorithm, each node in the network monitors the strength of the signal it receives from its neighbors. The node uses this information to compute the

signal stability or the likelihood that the signal will remain strong enough to maintain a reliable connection, for each potential route to the destination. The node then chooses the route with the highest signal stability as the best route. Signal stability-based adaptive routing algorithms are typically used in wireless networks, where the strength of the signal can vary due to factors such as distance, obstacles, and interference. These algorithms can help to improve the reliability and performance of the network by selecting routes that are less likely to be disrupted by changes in signal strength.

However, one potential downside of signal stability-based adaptive routing is that it may not always find the shortest or most efficient route to the destination. Because these algorithms prioritize signal stability over other factors, they may sometimes choose a longer or more circuitous route in order to maintain a strong signal. This can result in longer delays and higher overall network traffic.

Another potential downside of signal stability-based adaptive routing is that it relies on the availability of accurate and up-to-date information about the signal strength in the network. If the nodes in the network are not able to accurately measure or estimate the signal strength, the routing decisions may not be optimal. This can lead to poor performance and reliability in the network.

Additionally, signal stability-based adaptive routing algorithms can be computationally expensive, as they require each node to constantly monitor the signal strength and compute the signal stability for multiple routes. This can be a challenge in large or complex networks, where the number of nodes and routes may be very large.

13. **Zone Routing Protocol (ZRP):** ZRP is a proactive routing protocol that is designed to address the challenges of routing in large and complex wireless networks.

    In the ZRP, each node in the network is divided into several zones, or regions of influence, based on the strength of the signal it receives from its neighbors. The node maintains routing information only for the nodes within its zone and the nodes in the adjacent zones. This helps to reduce the amount of routing information that each node needs to maintain, and it makes the routing process more efficient and scalable.

    When a packet needs to be transmitted, the ZRP uses a combination of proactive and reactive routing to find the best route to the destination. If the destination is within the same zone as the source, the packet is sent directly to the destination using the precomputed route. If the destination is in a different zone, the packet is sent to the border node of the destination zone using the precomputed route, and then the border node uses a reactive routing algorithm to find the rest of the route to the destination.

    The ZRP is designed to be scalable and efficient, and it has been shown to perform well in large and complex wireless networks. It is also adaptive to changes in the network, as it can update the routing information and adjust to changes in the network topology over time.

    One potential downside of the ZRP is that it can be complex to implement and configure, as it involves dividing the network into zones and computing routing information for each zone. This can require significant computational resources, and it may not be suitable for all networks.

Another potential downside of the ZRP is that it relies on accurate information about the signal strength in the network in order to divide the network into zones and compute the routing information. If the nodes in the network are not able to accurately measure or estimate the signal strength, the routing decisions may not be optimal. This can lead to poor performance and reliability in the network.

Additionally, the ZRP is a proactive routing protocol, which means that it maintains routing information for the entire network in advance. This can be useful for maintaining efficient routes and adapting to changes in the network, but it can also result in a large amount of routing information that needs to be stored and maintained by each node. This can be a challenge in large or complex networks, where the amount of routing information can be very large.

14. Zone-Based Hierarchical Link State (ZHLS): ZHLS routing protocol is a hierarchical routing protocol that is designed to address the challenges of routing in large and complex wireless networks.

In the ZHLS protocol, the network is divided into a hierarchy of zones, with each zone being composed of several nodes. The nodes in each zone maintain routing information only for the nodes within their zone and the nodes in the adjacent zones. This helps to reduce the amount of routing information that each node needs to maintain, and it makes the routing process more efficient and scalable.

When a packet needs to be transmitted, the ZHLS protocol uses a combination of proactive and reactive routing to find the best route to the destination. If the destination is within the same zone as the source, the packet is sent directly to the destination using the precomputed route. If the destination is in a different zone, the packet is sent to the border node of the destination zone using the precomputed route, and then the border node uses a reactive routing algorithm to find the rest of the route to the destination.

The ZHLS protocol is designed to be scalable and efficient, and it has been shown to perform well in large and complex wireless networks. It is also adaptive to changes in the network, as it can update the routing information and adjust to changes in the network topology over time.

15. Core-Extraction Distributed Ad hoc Routing (CEDAR): CEDAR is a reactive routing algorithm that is designed to improve the performance and efficiency of routing in wireless networks.

In the CEDAR algorithm, each node in the network maintains a cache of routing information for the nodes that it has recently communicated with. When a packet needs to be transmitted, the node first checks its cache to see if it has a route to the destination. If it does, the packet is sent directly to the destination using the cached route. If it does not, the node uses a reactive routing algorithm, such as flooding or depth-first search, to find a route to the destination.

The use of a cache in the CEDAR algorithm allows it to take advantage of the spatial and temporal locality of communication in wireless networks. This means that nodes are more likely to communicate with the same nodes over time and that the nodes are more likely to be located near each other. By using cached routes, the CEDAR

algorithm can reduce the amount of routing overhead and improve the performance of the network.

The CEDAR algorithm is effective in improving the performance of wireless networks, and it has been used in several real-world applications. It is a reactive algorithm, which means that it does not maintain routing information for the entire network in advance, but it can adapt to changes in the network over time.

One potential downside of the CEDAR algorithm is that it relies on the availability of accurate and up-to-date routing information in the cache. If the cache is not properly maintained or if it becomes stale, the routing decisions may not be optimal. This can lead to poor performance and reliability in the network.

Another potential downside of the CEDAR algorithm is that it is a reactive algorithm, which means that it does not maintain routing information for the entire network in advance. This can be useful for reducing routing overhead and adapting to changes in the network, but it can also result in longer delays when a packet needs to be transmitted. This can be a particular challenge in networks with high traffic volumes or tight latency requirements.

Additionally, the CEDAR algorithm can be computationally expensive, as it requires each node to maintain a cache of routing information and perform routing calculations when a packet needs to be transmitted. This can be a challenge in large or complex networks, where the number of nodes and routes may be very large.

16. **Hierarchical state routing (HSR):** HSR is a routing protocol in which the network is divided into a hierarchy of domains, with each domain being composed of several nodes. The nodes in each domain maintain routing information only for the nodes within their domain and the nodes in the adjacent domains. This helps to reduce the amount of routing information that each node needs to maintain, and it makes the routing process more efficient and scalable.

    When a packet needs to be transmitted, the hierarchical state routing protocol uses a combination of proactive and reactive routing to find the best route to the destination. If the destination is within the same domain as the source, the packet is sent directly to the destination using the precomputed route. If the destination is in a different domain, the packet is sent to the border node of the destination domain using the precomputed route, and then the border node uses a reactive routing algorithm to find the rest of the route to the destination.

    Hierarchical state routing protocols are designed to be scalable and efficient, and they have been shown to perform well in large and complex networks. They are also adaptive to changes in the network, as they can update the routing information and adjust to changes in the network topology over time. Some examples of hierarchical state routing protocols include the High-Speed Routing (HSR) protocol and the Zone-Based Hierarchical Link State (ZHLS) routing protocol.

2.7 Vehicles communications

Intelligent Transportation System (ITS) is a technology that aims to provide a safe and secure travel experience by collecting, storing, and transmitting real-time traffic information to drivers. ITS allows for efficient road resource utilization and reduces energy consumption.

A cooperative communication system is one of the essential technologies in the framework of ITS. The term "cooperative" indicates the collaboration between vehicles and transport infrastructure using wireless networks. There are three types of communication in a cooperative intelligent transport system, vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I), and vehicle-to-everything (V2X).

- **Vehicle-To-Vehicle (V2V):** In these algorithms, information is transmitted among vehicles via Vehicle-To-Vehicle(V2V) connections using a mesh network and dedicated short-range communications to allow vehicles to communicate and share information about their speed, destination, intention, and surroundings and even real-time traffic conditions. This technology enables vehicles to send and receive omnidirectional messages, giving them a 360-degree "awareness" of nearby vehicles and therefore helps them determine potential crash threats in real-time. V2V communication enables many use cases, mainly improving driving safety or traffic efficiency and providing information or entertainment to the driver.

  Safety use cases of Vehicle-To-Vehicle includes warnings on entering intersections or departing highways, hazardous location warning(obstacle discovery, reporting accidents), sudden stop warnings(forward collision warning, pre-crash sensing or warning), lane change/keeping, warning assistance, privileging ambulances, fire trucks, and police cars.

  It can also offer multiple Efficiency applications such as Enhanced Route Guidance and navigation, intelligent intersections: adaptable traffic lights, automated traffic intersection control, green light optimal speed advisory, merging assistance: enters an on-ramp to a limited access roadway), and variable speed limits.

- **Vehicle-To-Infrastructure:** In Vehicle-to-Infrastructure, roadside infrastructure is also involved in information exchange. Infrastructure components include road signs, lane markings, traffic lights, and other intelligent road infrastructure connected via a wireless connection. Vehicle-to-Infrastructure have multiple safety applications that aim to decrease the number of accidents by predicting and notifying the drivers of information obtained through the communications between the vehicles and sensors installed on the road, such as warning for hazardous situations (like congestions, accidents, obstacles, etc.), merging assistance, intersection safety, speed management, rail crossing operations, priority assignment for emergency vehicles. Moreover, it can offer efficient applications to utilize roads and intersections better. These applications can operate locally at an intersection or a given road section. These applications include traffic jam notification, prior recognition of potential traffic jams, dynamic traffic light control, dynamic traffic control, connected navigation, and free parking space notification.

- **Vehicle-To-Everything:** V2X, also known as vehicle-to-everything, encompasses both V2V and V2I technology. V2X technology enables vehicles to communicate with the traffic system, including other vehicles and infrastructure. V2X can notify drivers of hazardous weather conditions, nearby accidents, and traffic congestion. A subset of V2X is Vehicle-to-Pedestrians(V2P), which can give warnings to pedestrians from their cellphones about vehicles driving erratically or at unsafe speeds in their immediate vicinity.

3 The problem of Vehicle overtaking

3.1 Perception challenges

Autonomous vehicles have made significant progress in various aspects of driving, including collision avoidance, cruise control, traffic congestion detection, lane following, and improved routing policies. These advancements have greatly contributed to the overall safety and efficiency of self-driving cars. However, one critical area where autonomous vehicles still face challenges is in performing safe overtaking maneuvers.

Overtaking maneuvers require the vehicle to assess the surrounding environment, identify appropriate gaps in traffic, and execute a maneuver that ensures both the safety of the occupants and other road users. Several factors contribute to the difficulties encountered by autonomous vehicles in this context, and they can be broadly classified into three categories: perception, algorithms, and safety assessment.

Perception plays a vital role in enabling autonomous vehicles to understand and interpret their surroundings accurately. To perform a safe overtaking maneuver, the vehicle must accurately perceive the position, speed, and intentions of other vehicles on the road. However, challenges arise due to the complexities involved in perceiving and tracking multiple objects simultaneously, especially in dynamic traffic scenarios. Factors such as occlusions, varying lighting conditions, and the presence of pedestrians or cyclists further complicate the perception task. Improvements in perception systems, including advanced sensor technologies such as LiDAR, radar, and cameras, along with robust object detection and tracking algorithms, are necessary to overcome these challenges.

3.2 Algorithmic Challenges

Algorithms are another crucial aspect that influences the capability of autonomous vehicles to perform safe overtaking maneuvers. The decision-making process for overtaking involves analyzing a multitude of factors, including the relative speeds of the surrounding vehicles, the available space for maneuvering, and the compliance with traffic regulations. Developing algorithms that can effectively balance these factors and make reliable decisions in real-time is a complex task. Overtaking maneuvers also require a clear understanding of the road geometry and the vehicle's dynamic constraints. Furthermore, algorithms need to consider the intentions and behavior of other road users to anticipate their reactions accurately. Continued research and development in algorithms, machine learning techniques, and artificial intelligence are necessary to enhance the decision-making capabilities of autonomous vehicles.

Safety assessment is of paramount importance in the development and deployment of autonomous vehicles. Overtaking maneuvers pose inherent risks, and ensuring the safety of the occupants and other road users is of utmost importance. Evaluating the safety of overtaking maneuvers involves assessing factors such as the available distance, time, and acceleration profiles required for executing the maneuver safely. Additionally, the assessment should consider the uncertainties associated with perception, prediction, and control systems. Establishing robust safety metrics, testing protocols, and validation procedures specific to overtaking maneuvers are crucial to address the challenges in this domain.

To overcome the difficulties faced by autonomous vehicles in performing safe overtak-

ing maneuvers, a holistic approach is required. This approach involves advancements in perception systems to accurately detect and track objects, development of sophisticated algorithms that can make reliable decisions in real-time, and establishing comprehensive safety assessment frameworks to evaluate and validate overtaking maneuvers. Collaboration between researchers, engineers, regulators, and industry stakeholders is essential to address these challenges effectively. By tackling these issues, autonomous vehicles can continue to evolve and improve their capabilities, eventually leading to safer and more efficient transportation systems.

3.3 Sensor Technologies

Perceiving the surrounding environment accurately is crucial for autonomous vehicles to navigate safely and make informed decisions. Sensors such as LiDAR (Light Detection and Ranging), cameras, radar, and GPS play a vital role in capturing data about the vehicle's surroundings. However, effectively utilizing and processing this information poses significant challenges, which are influenced by the specific objectives of the autonomous system.

LiDAR sensors emit laser beams to measure distances and create detailed 3D maps of the environment. They provide precise information about the position and shape of objects, enabling the vehicle to detect obstacles and navigate through complex scenarios. However, the high cost and limited range of LiDAR systems can present obstacles to widespread adoption. Overcoming these challenges requires advancements in sensor technology, such as the development of cost-effective LiDAR alternatives or improvements in the range and resolution of existing systems.

Cameras are another essential sensor used in autonomous vehicles. They provide visual information that helps with object detection, recognition, and understanding of the scene. However, processing camera data and extracting meaningful information from images can be computationally intensive. Sophisticated computer vision algorithms are necessary to analyze the visual data and make accurate interpretations. Additionally, challenges such as varying lighting conditions, occlusions, and image noise can affect the reliability of camera-based perception systems.

Radar sensors are widely used in autonomous vehicles for their ability to accurately measure distances and velocities of surrounding objects. They are particularly useful in adverse weather conditions, where other sensors may struggle. Radar systems excel at detecting and tracking moving objects, providing valuable information for collision avoidance and adaptive cruise control systems. However, radar sensors have limitations in resolving fine-grained details and can struggle with accurately detecting stationary objects or identifying the precise shape of an obstacle.

Global Positioning System (GPS) is commonly used for localization and navigation purposes. It provides accurate positioning information, which is essential for route planning and control algorithms. However, GPS signals can be affected by factors such as signal blockage in urban canyons or multipath interference. These limitations can lead to errors in position estimation and affect the overall accuracy of the autonomous vehicle's navigation system. Combining GPS with other sensor data, such as LiDAR or camera inputs, can help mitigate these limitations and enhance localization accuracy.

Developing optimal routing, navigation, and control algorithms for autonomous vehicles is a complex task. These algorithms need to process the sensor data, interpret

the environment, and plan the vehicle's actions accordingly. Factors such as real-time constraints, dynamic traffic scenarios, and safety considerations further complicate the algorithmic design process. Moreover, testing and optimizing these algorithms require extensive computational resources and often rely on state-of-the-art vehicles equipped with expensive hardware. Overcoming these challenges involves continuous research and development efforts to improve algorithmic performance, computational efficiency, and the availability of affordable testing platforms.

Therefore, perceiving the surrounding environment accurately using sensors like LiDAR, cameras, radar, and GPS is a significant challenge in autonomous vehicle development. Processing and interpreting this sensor data, along with developing optimal routing, navigation, and control algorithms, require continuous innovation and advancements. Additionally, the cost and availability of high-quality sensors and testing platforms play a crucial role in the development and optimization of these algorithms. By addressing these challenges, researchers and engineers can enhance the perception capabilities of autonomous vehicles, ultimately leading to safer and more reliable self-driving systems.

3.4 Safety Assessment Challenges

Evaluating the safety of an overtaking maneuver is a complex task that involves considering multiple elements and factors. These elements are crucial for determining whether it is safe for a vehicle to overtake another vehicle on the road. While human drivers rely on their perception and judgment to assess these factors, it can be challenging for them to gauge them accurately in real-time, making it impractical to solely depend on human capabilities for safety assessment. Autonomous vehicles, on the other hand, have the potential to overcome these limitations by leveraging advanced sensing technologies and sophisticated algorithms.

One of the key elements that need to be taken into account when evaluating the safety of an overtaking maneuver is the proximity and speed of oncoming vehicles. Assessing the distance and relative speed of oncoming vehicles is essential to determine whether there is enough time and space for the overtaking vehicle to safely pass the slower vehicle. Human drivers may struggle to accurately perceive and judge these parameters, particularly in situations where the oncoming vehicle is approaching at a high speed. Autonomous vehicles equipped with precise sensors, such as LiDAR and radar, can provide accurate measurements of the distance and speed of oncoming vehicles, enabling more reliable safety assessments.

The gap between the host (ego) vehicle and the leading vehicle is another critical factor in evaluating the safety of an overtaking maneuver. This gap determines the available space for the overtaking vehicle to maneuver and complete the passing maneuver without compromising safety. Human drivers may find it challenging to assess the adequacy of this gap accurately, leading to potential risks during overtaking. Autonomous vehicles, with their ability to accurately perceive and track multiple objects simultaneously, can precisely measure the distance between vehicles and calculate whether the gap is sufficient for a safe overtaking maneuver.

In addition to the proximity and speed of oncoming vehicles and the gap between vehicles, the relative distance of surrounding vehicles also plays a crucial role in safety assessment. Human drivers must consider the positions and movements of nearby vehicles

to anticipate any potential conflicts or collisions during an overtaking maneuver. The task of monitoring and assessing the movements of multiple vehicles can be cognitively demanding for human drivers, potentially leading to errors in judgment. Autonomous vehicles, equipped with advanced perception and prediction algorithms, can analyze the trajectories and behaviors of surrounding vehicles, providing a more accurate understanding of the traffic situation and enabling safer overtaking maneuvers.

By combining precise sensor data, advanced perception algorithms, and real-time analysis, autonomous vehicles can assess the safety of an overtaking maneuver more reliably than human drivers. However, it is important to note that the development and validation of these algorithms require extensive testing and evaluation under various scenarios to ensure their effectiveness and robustness. Safety assessment frameworks specific to overtaking maneuvers can be established to define safety metrics, evaluate performance, and verify the capabilities of autonomous vehicles in executing safe overtaking maneuvers.

In conclusion, relying solely on human capabilities to assess the safety of an overtaking maneuver can be impractical due to the challenges associated with accurately gauging factors such as the proximity and speed of oncoming vehicles, the gap between vehicles, and the relative distance of surrounding vehicles. Autonomous vehicles, equipped with advanced sensing technologies and sophisticated algorithms, have the potential to overcome these limitations and provide more reliable safety assessments. By leveraging precise sensors, real-time analysis, and comprehensive testing, autonomous vehicles can contribute to safer and more efficient overtaking maneuvers on the road.

3.5 Traffic Signal Absence Challenges

In real-world scenarios, traffic signals play a crucial role in guiding drivers and prohibiting overtaking in specific road areas. However, in many underdeveloped and developing countries, the absence or incorrect functioning of traffic signals creates a considerable obstacle. Without clear indications, drivers may be uncertain about when and where overtaking is allowed, leading to potentially unsafe maneuvers. This lack of consistent guidance poses challenges for the development of autonomous systems that aim to execute safe overtaking maneuvers.

Another critical problem that arises in the context of overtaking on rural roads is the non-compliance with traffic rules. In some regions, drivers may disregard or ignore traffic regulations, forcing others to take dangerous risks when attempting to switch lanes or overtake slower vehicles. This non-compliance with traffic rules further exacerbates the complexities of safe overtaking maneuvers, as the behavior of other drivers becomes unpredictable. Autonomous systems need to account for these unpredictable driving patterns and adapt their decision-making processes accordingly to ensure safe and efficient overtaking.

On two-lane roads, where the second lane is occupied by vehicles traveling in the opposite direction, assessing the safety of an overtaking maneuver becomes even more challenging. The autonomous system must accurately detect and track the positions and movements of oncoming vehicles while simultaneously evaluating the available space and time for the overtaking maneuver. The high speeds and acceleration typically associated with overtaking add to the complexity, as there is limited room for error. The system needs to be precise and dependable, analyzing real-time sensor data with minimal delay to ensure

the safety of the maneuver.

3.6 Adverse Weather Conditions

Bad weather conditions, such as rain, fog, snow, or glare, further add to the obstacles encountered during overtaking. These adverse weather conditions can reduce a driver's visibility and impair their ability to assess the safety of the maneuver accurately. Additionally, weather conditions can affect the performance of sensors like cameras, reducing their accuracy in perceiving the surrounding environment. However, by utilizing a suitable combination of sensors, such as LiDAR and radar, an autonomous driving system can still perform overtaking maneuvers safely under various driving conditions. These sensors can provide reliable distance and velocity measurements, allowing the system to make informed decisions even in challenging weather scenarios.

Addressing these challenges requires robust algorithms that can analyze and interpret complex real-time sensor data, adapt to varying driving conditions, and account for the unpredictable behavior of other drivers. Extensive testing and validation in diverse environments, including regions with different traffic cultures and weather conditions, are necessary to ensure the reliability and safety of the overtaking capabilities of autonomous vehicles.

In conclusion, the development of systems capable of executing safe overtaking maneuvers faces challenges related to the absence or incorrect functioning of traffic signals, non-compliance with traffic rules, the presence of oncoming vehicles on two-lane roads, and adverse weather conditions. Overcoming these challenges requires precise and dependable systems that can assess the safety of the maneuver accurately in real-time, adapt to unpredictable driving behavior, and perform reliably under varying weather conditions. By addressing these obstacles, autonomous driving systems can contribute to safer and more efficient overtaking on the roads, even in challenging real-world scenarios.

4 Related Works

4.1 The problem of traffic management

Numerous efforts have been undertaken in scholarly literature to tackle the issue of traffic routing by proposing and optimizing current algorithms. One proposed solution, as presented in [81], is a predictive model of future network states that employs Relevance Vector Machines (RVMs) to estimate traffic patterns in the near future. Another approach, as described in [82], involves using vehicle trajectory data to estimate traffic flow parameters at signalized intersections via a multi-layer Bayesian network (BN) and shockwave analysis (SA). Fuzzy logic rules are used to select a route for vehicles to avoid congestion in [83] and [84], whereas an adaptive ant colony algorithm using neural networks is proposed to solve routing problems in [85].

Moreover, traffic networks can be regarded as large graphs, and graph theory research offers various solutions to problems related to road networks [86]. In addition to graph partitioning, which offers a tremendous opportunity to solve the traffic routing problem by breaking down the network into multiple sub-networks, another promising approach is network partitioning. Partitioning significantly reduces memory usage and computation time for any routing algorithm and simplifies their analysis. Several graph partitioning

techniques have been proposed in the literature to address the problem of traffic networks. For instance, in [87] and [88], the proposed algorithms use natural cuts in the road networks to perform network clustering; natural cuts are dense regions typically connected by bridges, mountain passes, and ferries within a network. In [89], a graph partitioning algorithm based on a multilevel coarsening of the graph works by iterative bisections of the coarsened hypergraphs, starting from the smallest one. Additionally, in [90], multilevel graph partitioning is used to provide new effective crossover and mutation operators and combines them with scalable communication to enhance partitioning results calculation time. More multilevel graph partitioning techniques are proposed in [91].

The field of robotic routing and path planning abounds with algorithms capable of efficiently finding the most appropriate path in various environments, be it static or dynamic. In a static environment, the robot navigates around fixed obstacles, while in a dynamic environment, it must move and avoid independently moving obstacles or other robots.

Initially, graph search algorithms like Dijkstra's algorithm [92], A* algorithm [93], Lee's algorithm [94, 95], and the Bellman-Ford algorithm were utilized to find optimal solutions. However, these algorithms become increasingly computationally expensive and time-consuming as the search space grows.

Hybrid approaches have been developed to optimize these solutions. For example, in [96], the shortest path was found by combining the Bellman-Ford and Dijkstra algorithms. In [97], a hybrid method that combined the visibility graph algorithm with the A* algorithm was used to find the shortest collision-free path.

Recently, the traffic routing problem has been addressed using rapidly exploring random tree and some of its variations, such as RRT* and Real-Time RRT*[98]. Moreover, rule-based machine learning techniques, such as genetic algorithms[99, 100], artificial immune systems[101], and Swarm Intelligence [102, 103], have also been widely applied to find the optimal path for vehicles in a grid.

However, solving traffic routing problems can be challenging due to dynamic and random changes in traffic conditions, such as traffic lights, traffic flow, accidents, and weather. To address these challenges, algorithms require a deep revision. One approach is to incorporate time constraints, which results in the Time Constrained Shortest Path Problem (TCSPP). TCSPP restricts nodes in a graph to specific time windows, making them unavailable outside those windows. However, solving this problem is complex and challenging, as it was proven to be NP-complete in [104].

The management of traffic in densely populated urban areas is a subject of interest to both researchers and engineers. Several approaches have been proposed to find better solutions for handling vehicles at intersections. One method involves optimizing traditional traffic light signals using swarm intelligent algorithms [105] or providing real-time intersection status to drivers, allowing them to change their routes to avoid congested traffic [106]. Another approach is scheduling vehicle passage through intersections using various techniques, such as first-in-first-out, shortest remaining time, round-robin, fixed priorities, and multilevel queue scheduling [107–110]. A reservation-based approach has also been proposed [111]. Other solutions focus on traffic coordination based on vehicle trajectory rather than reserving the entire intersection [112]. However, these solutions are static and must be re-implemented for each intersection based on lane architecture. More advanced solutions have also been studied, such as changing road architecture by creating

bridges instead of intersections, which can be costly and not feasible in some areas.

4.2 The problem of vehicle overtaking

In the realm of safe passing predictions, several attempts have been made to develop systems for predicting safe overtaking maneuvers. One approach involved using a vehicle's navigation system to label road segments as safe or unsafe for overtaking based on road geometry, but it failed to consider real-time traffic in the second lane and still required human intervention. Another approach proposed a control function that integrated safety indicators such as time headway and time to collision to minimize collision risks. A different study utilized a path planning technique based on sigmoid functions for autonomous overtaking maneuvers, determining geometric curves based on the relative distance and speed of the leading vehicle. Additionally, a real environment representation was created using various onboard sensors to identify the vehicle's motion corridor to its destination and forecast the movement of other moving objects.

Fuzzy logic-based overtaking systems have also utilized camera sensors to make overtaking decisions. One system utilizes the distance between the leading and approaching vehicles as inputs to determine whether to overtake, approach the leading vehicle, or delay the overtake. Other fuzzy logic-based solutions can be found in [113] and [114].

Overtaking solutions have been explored using various machine learning techniques. For example, Double Deep Q-learning (DDQN) was used in [115] to learn a policy for longitudinal speed and lane change, while Deep Q-learning was applied in [116] to perform overtaking decision-making on one-way through two lanes. Other reinforcement learning-based approaches have also been attempted in [117], [118], and [119] to plan overtaking maneuvers. However, these techniques have only been verified using basic simulation techniques.

Researchers have investigated the potential of camera-based systems to enhance overtaking safety. For instance, in [120], a rear-mounted monocular camera was employed along with a Convolutional Neural Network (CNN) to detect and track overtaking vehicles for behavior analysis. Similarly, in [121], an image-based system was introduced that detects lane structure and vehicle presence to evaluate overtaking safety. Nevertheless, utilizing monocular vision-based systems alone may lead to a considerable false-positive detection rate, particularly when there is insufficient training data available.

To address this, a homogeneous sparse optical flow approach was proposed in [122] to classify the road image into dynamic and quasi-static regions, allowing for the detection of surrounding vehicles. However, these vision-based systems can be unreliable in real driving scenarios due to factors such as weather conditions like fog and rain.

Detecting oncoming vehicles during the overtaking maneuver is a highly complex challenge that can be affected by driving conditions and road architecture. Vision-based techniques [123] and machine learning approaches [124] have been used in numerous attempts to detect oncoming vehicles, but these methods often have limited effectiveness on curved roads. While a radar-based approach has been proposed for vehicle identification and tracking on curved roads [125], the high cost of radar and lidar-based systems remains a challenge.

To address the overtaking maneuver as a moving obstacle avoidance problem, search-

based algorithms such as rapidly exploring random trees (RRT) can be used. In [126], the authors defined a feasible maneuver in the longitudinal and lateral directions (accelerating or decelerating and changing lanes, respectively).

To enhance safety during overtaking, Vehicle-to-Vehicle (V2V) communication systems have been developed. For example, a collision warning system was developed in [127] that utilizes V2V communication. In [128], V2V communication was combined with fuzzy theory to control the parameters of a safety assistance system. A V2V wireless communication system was also studied in [129] and [130], where it was found to be effective in predicting unsafe overtaking maneuvers and warning the driver before the maneuver on rural highways.

Approaches based on Model Predictive Control (MPC), also known as Receding Horizon Control (RHC), have been extensively used in controlling dynamic systems, particularly in the domain of autonomous automotive. MPC is a powerful technique for handling multi-variable constrained control problems. In the literature, numerous attempts have been made to apply MPC to solve the overtaking problem.

One study proposed a path modeling approach to perform safe overtaking by generating overtaking trajectories using convex modeling on the lane's width and the preceding vehicle's length [131]. Another approach involved precomputing the state prediction using a four-wheel nonlinear dynamic model, stored in a lookup table, to plan collision-free trajectories [132]. A motion planner based on MPC was proposed in [133] to avoid other vehicles on the road by formulating the motion planning as a mixed-integer quadratic programming (MIQP) problem. The approach used lane-blocking obstacles (LBOs) and non-blocking obstacles (NBOs) constraints to implement in the boundary equations of the cost function of the MPC.

An MPC controller based on path velocity decomposition was developed in [134] to provide autonomous driving with collision avoidance using a time-scaled collision cone (TSCC). The forward velocity optimization layer was formulated as a convex quadratic programming (CQP) problem. In [135], MPC was used to determine the conflict probability of overtaking. The system used a dynamic model of the vehicles to estimate the future state of the vehicles.

One study proposed an overtaking control algorithm that accounted for the stochastic nature of traffic by predicting the position of surrounding vehicles using their preceding data [136]. In [137], a collision avoidance system was proposed that used deep learning combined with Model Predictive Control to perform vehicle maneuvers such as overtaking. The system estimated the probability of collision and minimized the vehicle's velocity in areas with high collision probability and increased it otherwise. Finally, a Deep Neural Network (DNN) model was proposed in [138] to make near human-like overtaking maneuver decisions such as vehicle overtaking.

Although the aforementioned research provides promising solutions for overtaking, most of them do not consider the actual dynamics of the vehicles, as they rely only on static and kinematic models of the vehicle and often ignore the vehicle's interaction with the road.

Learning-based approaches, such as those based on deep learning, can be viewed as a black box once they have been trained, making it impossible to fully understand their inner workings and without any safety guarantees. Furthermore, the probabilistic nature of

machine learning can result in unsafe system behavior [139]. In contrast, Model Predictive Control (MPC) approaches offer the ability to quickly integrate multiple constraints into a system, such as constraints related to obstacle avoidance, path following, speed, acceleration, and steering angle limitations. This results in a motion planning process similar to human driving, which results in a smoother and more comfortable driving experience. Additionally, MPC enables the implementation of hard constraints on control and states since natural constraints limit the inputs in every automation system, such as force, acceleration, speed, and angle of motion [140]. Furthermore, MPC provides the benefit of quickly adapting to changes and events in the environment.

5 Conclusion

The areas of traffic congestion, vehicle safety, and advanced traffic control have become essential focal points in the realm of autonomous vehicles. Although significant strides have been made in these areas, there remain certain aspects that necessitate additional research and development. To address these challenges, innovative systems have been proposed. For instance, the Multi-agent Traffic Coordinator, Multi-Cluster Q-routing, and vehicle overtaking systems are examples of proposed solutions aimed at improving traffic management and safety. By persisting in these endeavors, we can realize an intelligent and efficient transportation system that guarantees a seamless and secure experience for all individuals utilizing the road.

CHAPTER 2

Multi-agent traffic coordinator for A.Vs at intersections

1 Introduction

Intersections are a critical component of our transportation infrastructure, as they allow vehicles to cross paths and travel in different directions. However, when intersections become congested, they can cause a variety of problems for drivers. For example, when traffic is backed up at an intersection, drivers may be stuck waiting at red lights or stop signs for long periods, which can cause stress and frustration. Additionally, congestion at intersections can lead to increased fuel consumption and emissions, as cars are idling in traffic rather than moving efficiently through the intersection.

Many factors can contribute to congestion at intersections. For example, poor traffic flow design, a high volume of vehicles, or a lack of adequate traffic control systems can all lead to increased delays at intersections. Additionally, intersections located in urban areas with high population density are more likely to experience congestion, as there are more cars on the road in these areas.

To alleviate traffic congestion and reduce the negative impacts of intersections, transportation planners and engineers have developed a variety of solutions. Some of these include:

- **Intelligent transportation systems (ITS):** such as traffic signal optimization and real-time traffic management, can help to improve traffic flow at intersections and reduce delays.
- **Roundabouts:** Roundabouts are becoming increasingly popular as an alternative to traditional intersections. They are designed to reduce delays and improve safety by allowing vehicles to move continuously through the intersection without having to stop.

- ❑ **Alternative forms of transportation:** Encouraging the use of alternative forms of transportation, such as biking and walking, can help to reduce the number of cars on the road and alleviate congestion at intersections.
- ❑ **Urban design:** Urban design strategies, such as creating more compact and walkable neighborhoods, can help to reduce the need for car travel and decrease congestion at intersections.

In this section, we will propose a solution to address the problems of traffic congestion and safety at intersections by introducing a decentralized multi-agent system that utilizes deep reinforcement learning for the coordination of autonomous vehicles. This system aims to enhance safety, decrease traffic congestion and provide a more seamless and efficient transportation experience for drivers.

2 Preliminaries

2.1 Multi-agent systems

The field of artificial intelligence known as Multi-agent systems (MAS) [141] involves creating intricate systems consisting of multiple agents and the mechanisms required to coordinate their behavior. These systems are composed of several independent agents working together to solve complex problems beyond their individual capacities. One significant advantage of MAS is that each agent can be relatively straightforward, and implementation can be carried out independently, as long as the agents comply with the necessary specifications for interaction with others. This modular design enables easy scaling of MAS by incorporating additional agents, which is generally simpler than altering a larger system. Within a MAS, each agent can engage with its surroundings, take actions based on the environment's state, and receive rewards as feedback. This process of interaction is depicted in Figure 2.1.

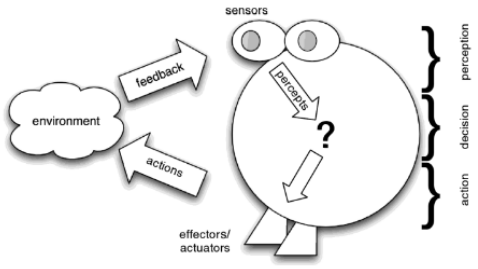


Figure 2.1: Illustrating the dynamic relationship between agents and their surroundings.

2.2 Deep Neural Network

Deep neural networks are a type of function approximator that has gained widespread popularity in recent years. This section provides an overview of the fundamental concepts behind these networks.

2.2.1 Basic structure

❑ **Linear Threshold Unit:** An LTU, also called an artificial neuron, takes several inputs and computes their weighted sum followed by a step or sign function. If the input exceeds the threshold, then the unit activates. Otherwise, it remains inactive. Figure 2.2 shows an LTU with the activation function is the step function.

$$sign(x) = \begin{cases} 1 & \text{if } x > \theta \\ -1 & \text{otherwise} \end{cases} \quad (2.1) \qquad step(x) = \begin{cases} 1 & \text{if } x > \theta \\ 0 & \text{otherwise} \end{cases} \quad (2.2)$$

$$z = w_1 x_1 + w_2 x_2 + ... + w_n x_n = w^T x \quad (2.3)$$

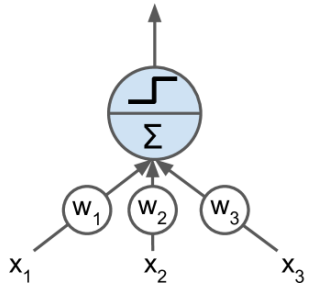


Figure 2.2: The linear threshold unit.

❑ **Perceptron:** The perceptron is a single layer of LTU; each neuron is connected to all the inputs. These connections are often represented using special passthrough neurons called input neurons. Moreover, an extra bias feature is generally added. This bias feature is typically represented using a special type of neuron called a bias neuron, which always outputs one. A perceptron can always learn to classify all training examples if training examples are linearly separable. However, it's not capable of simulating complex non-linear functions.

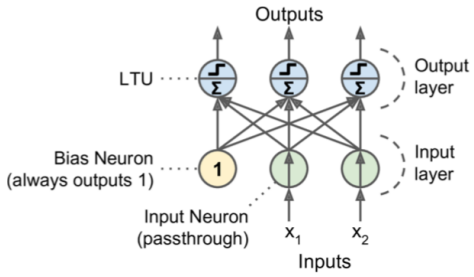


Figure 2.3: Perceptron diagram.

❑ **Multi-layer Perceptron and Deep Neural Network:** An MLP consists of one (passthrough) input layer, one or more layers of LTUs called hidden layers, and one final layer of LTUs called the output layer. Every layer except the output layer includes a bias neuron fully connected to the next layer. A deep neural network refers to a neural network that has more than two hidden layers.

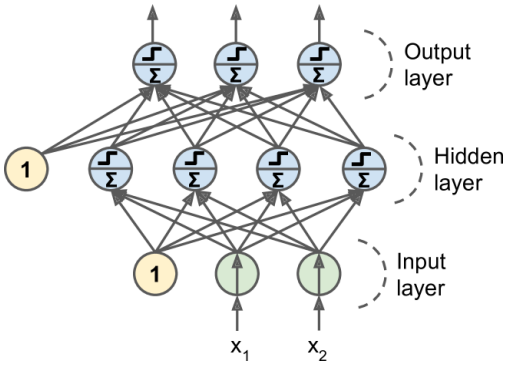


Figure 2.4: Basic structure of a Multi-Layer Perceptron.

2.2.2 Training a Deep Neural Network

The process of training a Deep neural network (DNN) involves modifying the connection weights of the neurons to reduce the error represented by the cost function. This is usually done by minimizing the difference between the model's predictions and the actual data in the training set using cross-entropy.

$$cost(y, \hat{y}) = -\sum_{j} y_j \log(\hat{y_j}) \tag{2.4}$$

Where $y \in [0, 1]$ is the ground truth value and $\hat{y}$ is the predicted probability. The objective is to make the prediction values and the ground truth as close as possible. Backpropagation is the mechanism by which the weights and biases that influence the output of a neuron are iteratively adjusted to reduce the cost function. The critical mathematical process involved in backpropagation is the calculation of derivatives. The backpropagation's operations calculate the partial derivative of the cost function with respect to the weights, biases, and previous layer activations to identify which values affect the gradient of the cost function.

A gradient-based backpropagation algorithm to train a DNN is as follows steps:

- Forward the input through the neural network to generate prediction $\hat{y}$
- Measure the cross-entropy error.
- Flow back the error's gradient to the lowest layer of the network.
- Modify the weights and bias based on the flowed gradient.

$$J(w_i) = cross_entropy(y_j, \hat{y_j}) = -\sum_{j} y_j \log(\hat{y_j}) \tag{2.5}$$

Where w_j is the weight of the DNN and $\hat{y}$ can be seen as a non-linear function of the weight.

Training a neural network with gradient descent requires the calculation of the gradient of the error function $J(w_j)$ with respect to the weights $w^k_{i,j}$ and biases b^k_i. Then, according

to the learning rate η, each iteration of gradient descent updates the weights and biases (collectively denoted θ) according to:

$$\theta^{t+1} = \theta^t - \eta \frac{\partial J(\theta^t)}{\theta} \tag{2.6}$$

2.2.3 Challenge of Training a Deep Neural Network

In training a deep neural network, we may face common challenges that hinder the model's performance.

❑ **Overfitting:** Overfitting occurs due to excessive training resulting in the model fitting exactly to the training set instead of generalizing over the problem; in this case will have low accuracy on the test dataset, although it has high accuracy on the training dataset.

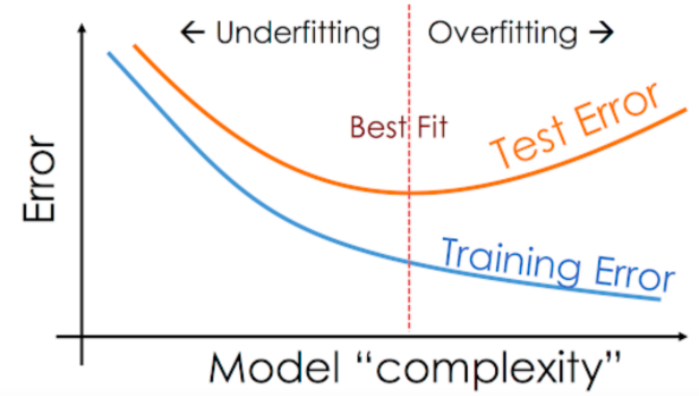


Figure 2.5: Model complexity and overfitting.

Different techniques are used to overcome the overfitting problem:

- ***Regularization:*** involves adding a penalty to the weight vectors in the cost function to either limit their magnitude or quantity, to prevent overfitting and creating overly complex models.
- ***Early stopping:*** It involves training the model until the validation error starts to increase and then stopping the training process. This ensures that the model does not continue to fit the training data after it has started to overfit.
- ***Dropout:*** Randomly dropping some neurons during training helps to simplify the model and increase its sparsity, by disregarding certain neurons.
- ***Data augmentation:*** By augmenting the data through modifications such as changing angles, illumination, saturation, or exposure, the size of the data set is increased, making it harder for the model to overfit due to its larger size.
- ***Cross-validation:*** Splits the data into multiple sets and trains the model on each set individually; this allows the model to be trained on different data and prevents it from being overfitted to a particular subset of data.

❑ **Vanishing/Exploding Gradient:** The back-propagation algorithm backflow the error gradient from the output to the input. The gradient will often get smaller and smaller and approach zero, eventually leaving the weights of the initial or lower layers nearly unchanged. As a result, the gradient descent never converges to the optimum; this is known as the vanishing gradients problem. On the contrary, the gradients will get larger in some cases; this, in turn, causes large weight updates and causes the gradient descent to diverge; this is known as the exploding gradients problem. Several techniques are used to overcome the vanishing/exploding gradient problem:

- ***Non-saturating activation function:*** The sigmoid activation function has a limiting effect on larger inputs, both positive and negative, making it unsuitable for use in the hidden layers of neural network. Using the Rectified Linear Unit (ReLU) activation function instead of the sigmoid function facilitates the training process. The gradient of the sigmoid function is minimal when it approaches -1 or 1, causing it to saturate. ReLU, defined as $\max(0, z)$, has a larger gradient. However, the problem of dying ReLU arises when some neurons stop producing any output other than zero. The leaky ReLU can overcome this problem and speed up training, with $LeakyReLU = \max(az, z)$, where "a" is the slope for values of "z" less than 0.
- ***Gradient clipping:*** Clip the gradients during backpropagation so that they never exceed some threshold.
- ***Batch Normalization:*** Batch normalization is a technique to address the problem that the distribution of each layer's inputs changes during training as the parameters of the previous layers change. The operation zero-centers and normalizes each input, then scales and shifts the result. Batch normalization lets the model learn the optimal scale and mean of each layer's inputs.

$$\mu_B = \frac{1}{m_B} \sum_{i=0}^{m_B} x^{(i)} \tag{2.7}$$

$$\sigma_B^2 = \frac{1}{m_B} \sum_{i=0}^{m_B} (x^{(i)} - \mu_B)^2 \tag{2.8}$$

❑ **Local Minimum:** A local minimum is a problem where the model gets stuck because of a small learning rate and can be solved using the followings:

- ***Momentum:*** Momentum is a property where objects maintain their movement. In training, the use of momentum helps to overcome local minima by allowing the gradient to keep moving instead of getting stuck. During each iteration, a momentum term is added to the gradient, calculated by multiplying the local gradient by a momentum variable, resulting in a momentum vector.
- ***RMSProp:*** Root Mean Square Propagation is a technique that addresses oscillations in a different way than momentum. Unlike momentum, RMSprop adjusts the learning rate automatically, by having a smaller rate for parameters that change frequently, and a larger rate for others. This eliminates the need for manual tuning of the learning rate.

- ***Adam:*** RMSProp and Momentum take contrasting approaches. While Momentum accelerates our search in the direction of minima, RMSProp impedes our search in the direction of oscillations. Adam or Adaptive Moment Optimization algorithms combine the heuristics of both Momentum and RMSProp.

2.3 Reinforcement learning

Reinforcement learning involves maximizing a numerical reward signal by learning how to map situations to actions without being explicitly told which actions to take. The agent discovers which actions result in the highest rewards by trying different actions. This type of machine learning model is trained to make a sequence of decisions in real-time to achieve a goal in an uncertain and potentially complex environment. In more complex scenarios, the actions taken not only impact the immediate reward but also the next situation and all subsequent rewards. The two key distinguishing features of reinforcement learning are trial-and-error search and delayed reward.

A typical reinforcement learning model links an agent's perception and behavior to its surroundings. A scalar reinforcement signal conveys to the agent the state transition value in response to the action. The agent's behavior should tend to raise the long-run total of reinforcement signal levels as a consequence of systematic trial and error, guided by a wide range of algorithms Figure 2.6.

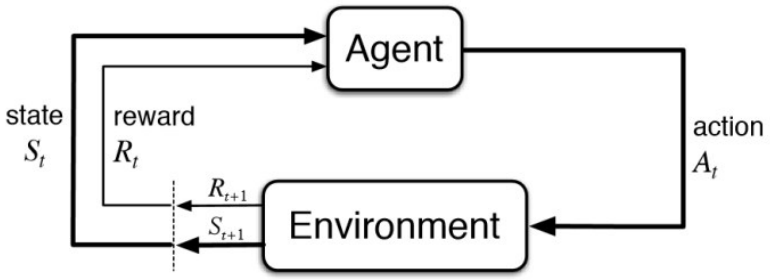


Figure 2.6: Standard model of reinforcement learning.

Reinforcement learning and supervised learning are distinct approaches in the field of machine learning. In supervised learning, which is the focus of most current research, the system learns from a training set of labeled examples provided by an external supervisor who is knowledgeable about the correct actions for each situation. Each example consists of a description of a situation and a label specifying the correct action to take, typically involving the identification of a category to which the situation belongs. The aim of supervised learning is to enable the system to generalize its responses and correctly act in situations that were not included in the training set.

RL algorithms can be mainly divided into two categories model-based and model-free:

❑ **Model-based:** They involve an agent that tries to comprehend its surroundings and builds a model of it based on how it interacts with them. The preferences of a greedy agent will always try to carry out an action that will get the maximum reward, regardless of what that action may cause. In such a system, preferences precede the effects of the actions.

❑ **Model-free:** Through the use of algorithms like Policy Gradient, Q-Learning, SARSA, and others, model-free algorithms try to learn from experience the effects of their actions. In other words, such an algorithm will repeatedly execute the same action and, based on the results, modify the policy (the plan that guides its actions) to maximize rewards.

A model-based approach may be the ideal choice for playing chess or a robotic arm in an assembly line of a product where the environment is static, and completing the task as quickly as possible is our primary concern. However, when it comes to real-world applications like self-driving cars, a model-based strategy might encourage the vehicle to strike a pedestrian in order to get there faster (maximum reward), but a model-free strategy would require the vehicle to wait until the road is clear (optimal way out).

Beyond the agent and the environment, we can identify four main sub-elements of a reinforcement learning system: a policy, a reward signal, a value function, and, optionally, an environment model.

❑ **Policy:** A policy defines the learning agent's behavior at a given time. Roughly speaking, a policy is a mapping from perceived states of the environment to actions to be taken in those states. It corresponds to what in psychology would be called a set of stimulus-response rules or associations. In some cases, the policy may be a simple function or lookup table, whereas, in others, it may involve extensive computation, such as a search process. The policy is the core of a reinforcement learning agent in the sense that it alone is sufficient to determine behavior. In general, policies may be stochastic, specifying probabilities for each action.

❑ **Reward function:** The objective of a reinforcement learning problem is defined by a reward function. The environment sends a reward to the reinforcement learning agent on each time step. The reward function maps each perceived state or state-action pair of the environment to a single number that specifies the intrinsic desirability of that state. This allows the agent to make conclusions instead of predictions. The agent's sole aim is to maximize the total reward it receives over the long run, and the reward signal determines what constitutes good and bad events for the agent. In biological systems, rewards are analogous to experiences of pleasure or pain and are thus the defining features of the problem faced by the agent. The reward signal is the primary basis for altering the policy, such that if a low reward follows an action selected by the policy, the policy may be modified to choose a different action in that situation in the future. Reward signals may generally be stochastic functions of the environment state and actions taken.

❑ **Value function:** A value function specifies what is good in the long run. i.e., the value of a state is the total amount of reward an agent can expect to accumulate over the future, starting from that state. Whereas rewards determine the immediate, intrinsic desirability of environmental states, values indicate the long-term desirability of states after considering the states that are likely to follow and the rewards available in those states. For example, a state might always yield a low immediate reward but still have a high value because it is regularly followed by other states that yield high rewards. And the reverse could be true. To make a human analogy rewards are

somewhat like pleasure (if high) and pain (if low), whereas values correspond to a more refined and farsighted judgment of how pleased or displeased we are that our environment is in a particular state.

❑ **Environment model:** A model is a system that imitates the behavior of the environment or enables inferences about how the environment will act. For instance, when given a state and action, the model can anticipate the resulting next state and reward. Models are utilized for planning, which refers to any approach for selecting a course of action by contemplating potential future scenarios before encountering them.

The Markov decision process (MDP), regarded as the standard when formalizing problems regarding learning sequential decision-making, is frequently used to model reinforcement learning problems. In MDP, the states are treated as a Markov chain, and the agent chooses the transitions (actions) based on the rewards r provided by the environment at each time step t.

2.4 Q-learning

An early advancement in reinforcement learning was the creation of Q-learning, a type of off-policy Temporal-Difference control algorithm. It is part of a category of solutions known as tabular methods(Figure 2.8). Q-learning is a model-free approach that can find a nearly ideal strategy for any finite Markov decision process, making it ideal for repeated interactions with unknown opponents. This method does not require a full understanding of the environment, instead, it learns by directly observing its dynamics. Additionally, it updates its predictions based on previous estimates, rather than waiting for the outcome of the process.

Q-learning is defined by:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)] \tag{2.9}$$

In Q-learning, the action-value function, represented by Q, is directly approximated by the optimal action-value function, represented by Q*. This simplifies the analysis of the algorithm and allows for early convergence proofs. The policy still plays a role in determining which state-action pairs are visited and updated, but as long as all pairs continue to be updated, correct convergence is guaranteed.

Q-learning relies on trial and error. The agent or controller chooses an action or control output at each state based on the Q-values of each action. The Q-value for a specific state-action combination, represented by Q(s,a), is the expected discounted sum of future payoffs that result from taking action a from state s and following an optimal policy thereafter. Once these values are learned, the optimal action from any state is the one with the highest Q-value.

Q-values can be thought of as the quality of an action taken from a specific state. The policy still impacts which state-action pairs are visited and updated, but no more than that; this is why Q-learning is often referred to as off-policy Temporal-Difference (TD) learning.

The Q-learning algorithm is represented in Figure 2.7.

Algorithm 1: Epsilon-Greedy Q-Learning Algorithm

Data: α: learning rate, γ: discount factor, ϵ: a small number
Result: A Q-table containing Q(S,A) pairs defining estimated optimal policy π^*

```
/* Initialization                                                    */
```
Initialize Q(s,a) arbitrarily, except Q(terminal,.);
Q(terminal,.) $\leftarrow$ 0;
```
/* For each step in each episode, we calculate the
   Q-value and update the Q-table                                    */
```
for *each episode* **do**
```
   /* Initialize state S, usually by resetting the
      environment                                                    */
```
   Initialize state S;
   for *each step in episode* **do**
      do
```
         /* Choose action A from S using epsilon-greedy
            policy derived from Q                                   */
```
         A $\leftarrow$ SELECT-ACTION(Q, S, ϵ);
         Take action A, then observe reward R and next state S';
         Q(S, A) $\leftarrow$ Q(S, A) + α [R + $\gamma \max_a$ Q(S', a) - Q(S, A)];
         S $\leftarrow$ S';
      while *S is not terminal*;
   end
end

Figure 2.7: Q-learning algorithm

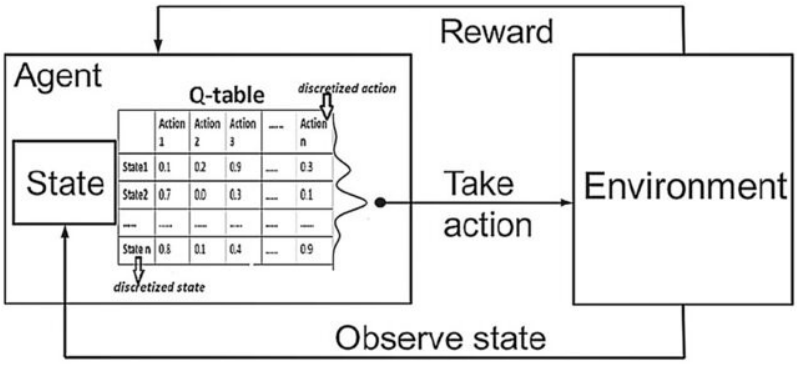


Figure 2.8: Q-learning [142]

2.5 Deep reinforcement learning

With the increasing capabilities of computational technology in recent years, it has become possible to use deep neural networks to tackle reinforcement learning problems, resulting in the emergence of deep reinforcement learning. This field aims to bring together the advancements in deep neural networks and reinforcement learning algorithms to develop agents that can behave intelligently in complex situations without the need for human input. There are various methods for deep reinforcement learning, such as Deep Q Learning (DQN), Double DQN, and Dueling DQN, but this discussion will focus solely on deep Q-learning.

In deep Q-learning [143], a value function $Q(s, a)$ is estimated over the learning process.

A Q-value represents the expected sum of discounted future rewards r_t the agent expects to receive by executing an action a from a state s [144].

$$Q(s_t, a_t) = \mathbb{E}[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ...|s, a] \tag{2.10}$$

Q-value functions decompose into a Bellman equation:

$$Q(s, a) = \mathbb{E}_{a',b'}[r + Q(s', a')|s, a] \tag{2.11}$$

The deep Q-learning algorithm uses a deep neural network (Figure 2.9)as a function approximator for the Q-value. In training, we observe a transition from a state s to state $s^{'}$ by choosing the best action a and receiving a reward r to estimate the error.

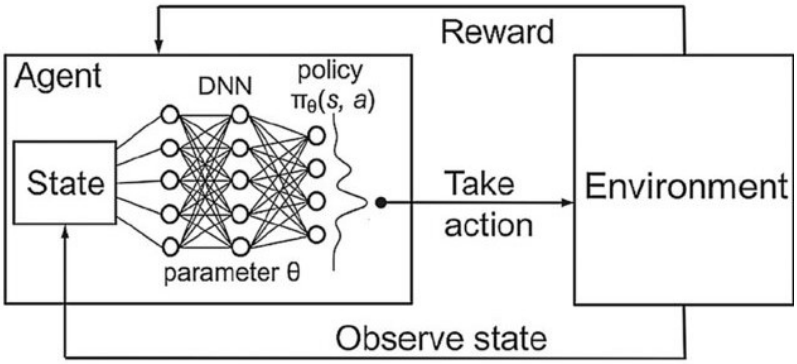


Figure 2.9: Deep Q-learning [142]

$$Error = [r + \gamma max_{a'} Q(s', a') - Q(s, a)]^2 \tag{2.12}$$

Given a transition $< s, a, r, s' >$, the Q action-value calculation process is detailed as follow:

1. Feed-forward the current state s to get predicted Q-values for all actions.
2. Choose an action a using ϵ-greedy to get into the next state s'
3. Feed-forward state s' and calculate maximum overall network outputs $\max_{a'} Q(s', a')$.
4. Set the Q-value target with the new value for the chosen action.

$$Q(s, a) = r + \gamma max_{a'} Q(s', a') \tag{2.13}$$

5. Update network weights using back-propagation.

3 Proposed system

3.1 System architecture

The multi-agent system proposed in this research consists of two types of agents: a vehicle agent installed in each vehicle and an intersection agent installed at each intersection. These agents communicate with one another to coordinate traffic at specific intersections, as shown in Figure 2.10. Each agent has its specific responsibilities and architecture, which are described as follows:

- **Vehicle agent (VA):** installed in each car that travels through an intersection. It includes a communication system that allows it to receive and send messages to and from the intersection agent (IA). This communication can be done through various methods such as radio signals, GSM, WiFi, or 4G, as long as it is able to work accurately with moving agents at high speeds. The VA can use different techniques such as sensors or GPS to detect when it is approaching an intersection and initiate a connection. In this research, we have chosen to use a GPS-based method for detecting intersections, but other approaches could also be effective. The VA also has a mechanism for controlling the vehicle's speed and brakes based on the IA's response and instructions.

- **Intersection agent (IA):** A pre-trained intelligent agent designed to oversee traffic flow for autonomous vehicles at intersections. Its primary objective is to determine the most suitable action (such as accelerating, decelerating, or maintaining current speed) to facilitate the smooth and secure passage of vehicles through the intersection, avoiding collisions with other vehicles. The essential knowledge that the IA possesses is shared among IA's with a similar architecture (number and geometry of lanes), which eliminates the need for retraining and enables them to share their experiences. The IA also takes into account the type of vehicles on the road, giving priority to special vehicles such as ambulances and police cars.

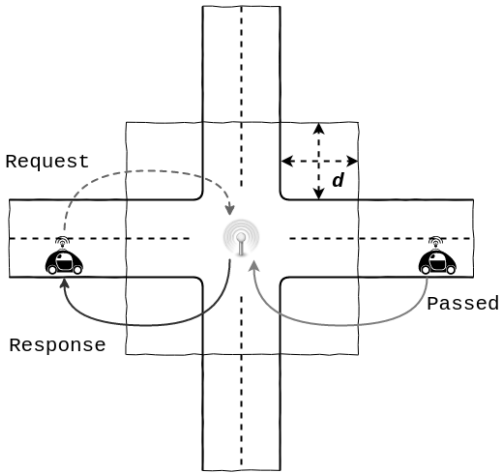


Figure 2.10: Proposed system Components' overview.

The exchange of messages between the intersection agent and the vehicle agent follows a specific process, as illustrated in Figure 2.11. This process involves multiple steps, including the initial transmission of a message from the IA to the VA, the receipt and processing of the message by the VA, and the subsequent response from the VA to the IA. These steps are essential for coordinating traffic at intersections and ensuring the smooth and safe passage of vehicles through the intersection.

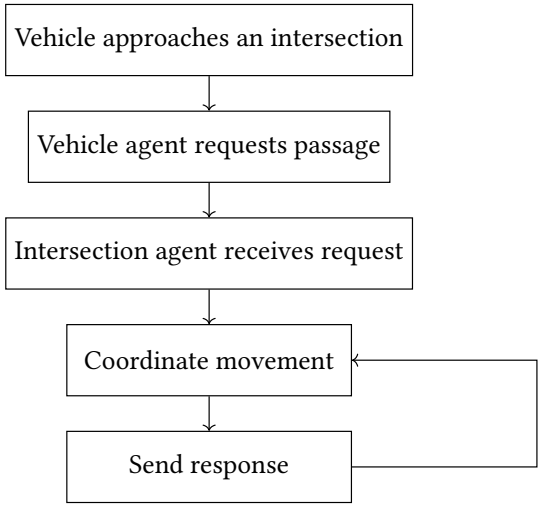


Figure 2.11: Proposed system algorithm.

1. As a vehicle approaches an intersection, it establishes a connection with the Intersection Agent (IA) at a predetermined constant distance "d" and remains connected until it exits the intersection. The assumption is that the Vehicle Agent (VA) will follow the IA's commands without deviation.

2. Upon receiving a request, the Intersection Agent establishes a connection with the Vehicle Agent and generates commands to safely guide the vehicle through the intersection, avoiding any collision with other vehicles. If a safe passage is not available, the IA will instruct the VA to decelerate and wait for further instructions.

3. Once a connection is established with the Intersection Agent, the vehicle surrenders full control of the steering wheel, throttle, and brakes to the IA.

4. After the vehicle passes the intersection, the IA relinquishes control, and the driver takes full control of the vehicle.

To minimize message size and reduce transfer delay and data loss, the communication between the Vehicle Agent (VA) and Intersection Agent (IA) is kept concise. The messages contain only essential information.

When a vehicle requests passage through an intersection, it sends information about its identity, position, destination (left turn, right turn, going straight), original speed, dimensions (width and length), and priority at the intersection (normal, police car, ambulance, etc.) to the IA. If the IA does not respond within a specified time interval "t," the vehicle

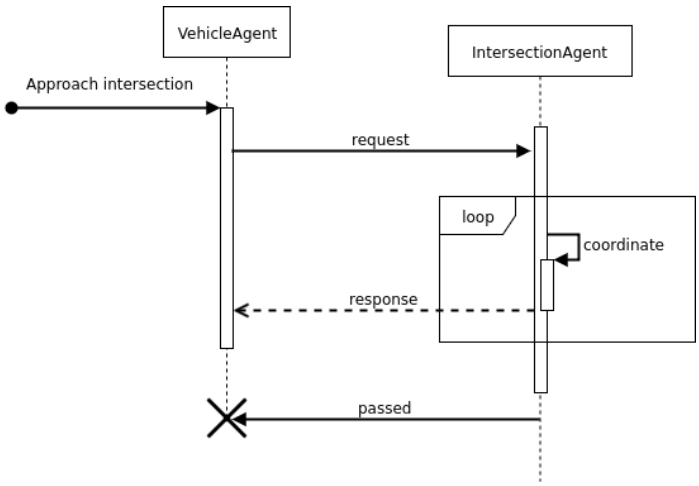


Figure 2.12: Exchange of information between the intersection agent and the vehicle agent.

resends the request until it receives a response. The vehicle is not permitted to enter the intersection until it receives a response. Examples of the message formats for the request and response exchanged between the VA and IA are shown in Listing 2.1 and Listing 2.2, respectively.

```
<?xml version="1.0" encoding="UTF-8"?>
<xml>
    <type value="Request" />
    <vehicleId>110</vehicleId>
    <position>
        <x>...</x>
        <y>...</y>
    </position>
    <destination value="left" />
    <speed>50</speed>
    <size>
        <width>...</width>
        <lenght>...</lenght>
    </size>
    <priority>normal</priority>
</xml>
```

Listing 2.1: Request message

```
<?xml version="1.0" encoding="UTF-8"?>
<xml>
    <type value="Response" />
    <vehicleId>110</vehicleId>
    <action>accelerate</action>
</xml>
```

Listing 2.2: Response messsage

3.2 System inputs

The Intersection Agent (IA) utilizes a set of features from both the current vehicle (v_c) and other vehicles present at the intersection to make decisions. These features include continuous values for position in the x and y direction, speed, width, and length, as well as discrete features such as destination and priority. These features are organized into a matrix with dimensions $m \times 5$, where m represents the maximum number of vehicles that can be handled simultaneously. If the number of current vehicles is less than m, the remaining cells are padded with zeros. This matrix is then flattened into a 1D array, where each cell corresponds to an input node in the neural network's input layer. The layout of these inputs is shown in Figure 2.13.

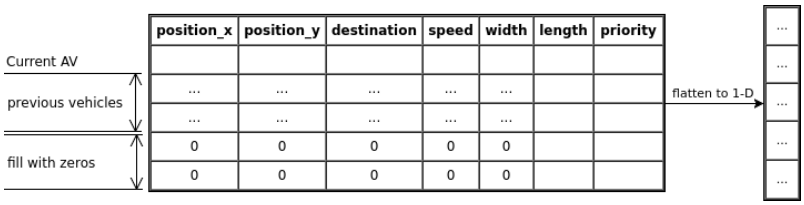


Figure 2.13: Vehicles arrangement to input to the deep neural network.

3.3 Deep neural network architecture

To create a deep neural network we used a neural network with twenty fully connected hidden layers and a Rectified Linear Unit (ReLU) [145] Equation 2.14 as an activation function. The choice of the activation function has been discussed in Equation 2.14. Each layer has 300 nodes.

$$f(x) = max(0, x) \tag{2.14}$$

3.4 System output

The action space of the deep neural network used in this system consists of three actions: *keep_same_speed*, *accelerate* and *decelerate*. At each time incrementation *t* the Intersection Agent must instruct the Vehicle Agent to execute one of these actions. The chosen action is determined by selecting the output with the highest Q-value from the neural network, as shown in Figure 2.14

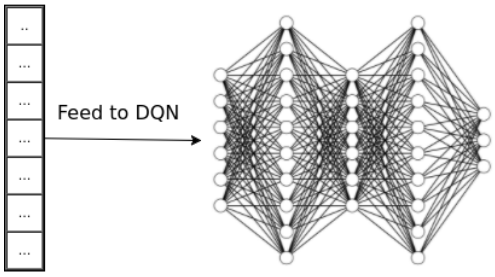


Figure 2.14: Vehicles arrangement to input to the deep neural network.

3.5 Reward function

While training the agent, a mechanism is required to evaluate each action and maximize its expected discounted future rewards for state-action pairs. When designing the reward function, we placed particular emphasis on passing with the highest speed and giving priority to vehicles with higher priority.

$$r = \begin{cases} speed * priority & if \quad speed > 0 \quad collision = false \\ 0.1 & if \quad speed = 0 \quad collision = false \\ 0 & if \quad collision = true \end{cases} \tag{2.15}$$

4 Results and Discussion

In order to evaluate the effectiveness of the proposed system, we utilized Python to develop a simulation environment. The simulation involves the coordination of traffic within a demonstration intersection using the method outlined earlier.

The simulation environment allows us to control the movements of a single vehicle (represented by the color green) as it traverses the intersection from left to right, while the movements of the other vehicles (represented by the color red) are randomly generated to simulate real-world behavior. Please refer to Figure 2.17.

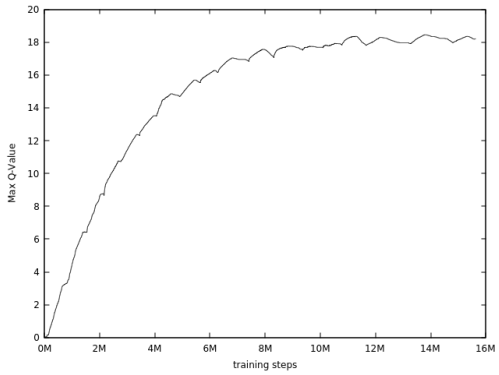


Figure 2.15: Estimated Max Q-value.

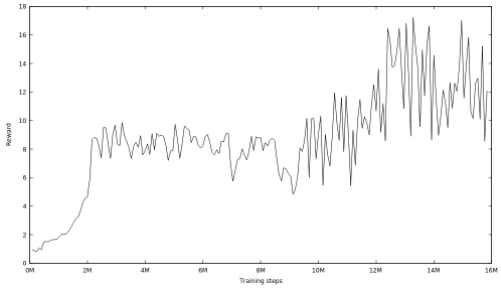


Figure 2.16: Received reward.

As we executed the simulation program, we monitored the Q-value and the rewards obtained by the agent. Please refer to Figure 2.15 and Figure 2.16.

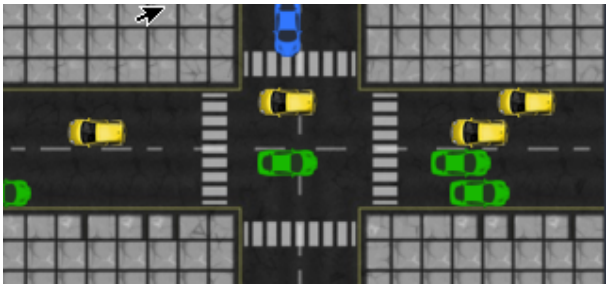


Figure 2.17: Simulation screen-shot.

5 Conclusion

In this chapter, we presented an overview of current solutions for addressing traffic congestion at intersections. However, many of these solutions are either fixed or require customization for different intersection configurations, depending on the number and width of lanes.

To overcome these limitations, we developed a multi-agent system that leverages deep neural networks and reinforcement learning to coordinate traffic flow for autonomous vehicles at intersections. This approach enables rapid adaptation to any new intersection design, without the need for additional implementation efforts. Additionally, it enables safe and efficient passage through intersections, without the need for human intervention.

Nevertheless, there are still some challenges to overcome, such as accounting for pedestrian crosswalks and non-autonomous vehicles. Moreover, there are opportunities for further enhancements, such as predictive vehicle routing based on destinations and communication between intersection systems to reduce the number of vehicles approaching an intersection simultaneously.

CHAPTER 3

Multi-Cluster Q-routing for traffic networks

1 Introduction

As previously noted, previous studies have suggested efficient and optimal routing methods for traffic networks, but they often struggle to perform well in large real-world networks. These issues stem from the exponential increase in computational and memory demands, particularly when routing must be done in real-time. Simple optimizations and algorithm enhancements have proven ineffective. To address this, we propose a multi-cluster Q-routing model, which partitions the network into smaller clusters and uses Q-routing as the protocol for routing inner and outer traffic for each cluster. By separating inner-cluster traffic from outer-cluster traffic, the model reduces the overall processing power required.

The proposed system utilizes Q-learning algorithms and spectral clustering to provide a decentralized, dynamic, and proactive routing solution. Its main benefit is the reduced computational needs as the routing process is distributed among multiple smaller nodes. Additionally, it can quickly adapt to changes in the network topology by efficiently propagating routing information among adjacent nodes. Furthermore, it offers real-time routing by using pre-calculated path weights that reflect the current state of each path in the network.

2 Preliminaries

2.1 Q-routing

Q-routing is a reinforcement learning algorithm based on the Q-learning approach to determine the optimal route to follow. Each node in the network has an intelligent agent that can dynamically determine the optimal path to each destination. In Q-routing, the policy π^* is progressively improved and updated using local information and information communicated from adjacent nodes. Unlike traditional routing algorithms that require global routing information that is generally unobtainable at the route calculation time and can only be obtained until the destination is reached.

In Q-routing [146], at each intersection, an intelligent agent can manage the intermediate routing of incoming vehicles to their destination.

Each routing node propagates its internal information, directly connected links, and the state of those links to all adjacent nodes. This way, via real-time state propagation, each routing node maintains a full network state awareness by updating itself incrementally whenever a change occurs in the network.

Each intersection has to determine the vehicle's best-remaining route to its final destination based on a policy π^*. In other words, the intersection has to determine the best next node or set of nodes the vehicle has to go through to reach its destination in an optimal time *t*. In Q-routing, all decisions are made using only local information about the network state stored as a table of Q-values.

Let $Q_x(d, y)$ be the estimated total travel time from a source node x to a destination node d by taking one of the immediate neighbors y of x, including any additional time the vehicle has to wait at traffic lights at the intersection x. In fact, in Q-routing, an intersection will always choose the neighbor with the minimum estimation t calculated using Equation 3.1 to route traffic into the next node decided using the result of the Equation 3.2. Where $z \in N(y)$ is the set of all immediate neighbors of intersection y.

$$t = \min_{z \in N(y)} Q_x(d, z) \tag{3.1}$$

$$y = \arg \min_{z \in N(y)} Q_x(d, z) \tag{3.2}$$

After routing a vehicle to an intersection, the node x updates its Q-value $Q_x(d, z)$ using the propagated estimation by applying the following equation:

$$Q_x(y, d) = Q_x(y, d) + \eta(q + s + t - Q_x(y, d)) \tag{3.3}$$

Where η is the fixed learning rate, q is the time spent in the traffic lights of node x, and s is the travel time between nodes x and y.

The essential part of Q-routing is that the traffic state is propagated from one node x to its next neighbor y only as a response to traffic being transmitted via that vertex as described in Figure 3.1. In fact, as the size of the network increases, the propagation of the state of the network slowdown significantly, resulting in one of the biggest problems in Q-routing.

2.2 Graph clustering

Graph clustering involves the grouping of vertices in a network into multiple homogeneous clusters or sub-networks, while taking into account the edge structure of the graph. This approach maximizes the number of edges within each cluster and minimizes the number of edges between clusters. The resulting clusters are disjoint subsets of the dataset, such that the data within each cluster are more similar to each other than to the data in other clusters. To perform effective graph clustering, it is necessary to identify elements with similar properties by detecting patterns and underlying structures within the network. The

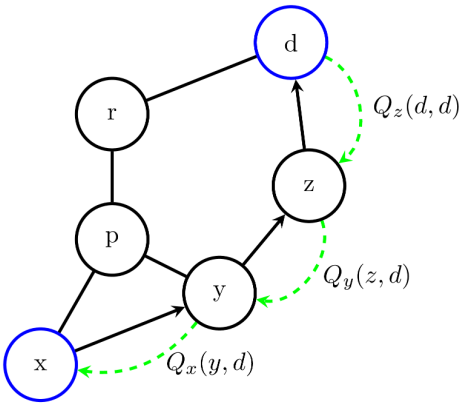


Figure 3.1: Example of Q-routing (solid arrows represent vehicles being routed and dashed arrows represent traffic state propagation).

similarity between nodes is typically calculated using topological criteria. Therefore, the role of clustering is to uncover natural structures within the dataset.

Cluster analysis is not only a valuable cognitive tool but also a method for reducing large datasets by replacing a group of data with a compact representation, such as the center of gravity of the group.

Similarity-based clustering is a technique that partitions data based on the similarities between pairs of points. This approach is applicable to community detection in social networks, where nodes represent individuals connected by links, and the "strength" of a link is determined by the frequency of interaction with other nodes.

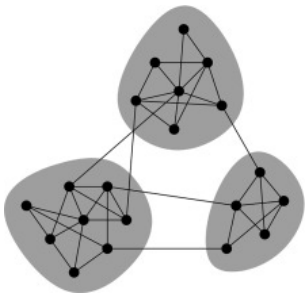


Figure 3.2: Graph clustering example.

Many algorithms have been suggested for graph clustering, such as k-means, Iterative Conductance Cutting, Geometric MST Clustering, and Markov Cluster Algorithm. One of the most used approaches in unsupervised machine learning is spectral clustering.

2.3 Spectral Clustering

Spectral clustering is a mathematical approach that merges many aspects of both linear algebra and graph theory. By identifying low-cost graph cuts, spectral clustering performs the grouping of clusters using the information from the eigenvalues (spectrum) and eigenvectors of the Laplacian matrix.

The power of spectral clustering relies on its implementation simplicity and efficient execution, as the partition can be obtained simply through the spectrum of its associated graphic matrix M.

Let $G = (V, E)$ be an undirected graph of vertex set $V = \{v_1, .., v_n\}$. We define the weighted adjacency matrix W of the graph G whose elements $w_{ij} = w_{ji} > 0$ represent the strength of the connection between i and j. If the nodes i and j are not connected, we set $w_{ij} = 0$, and we assume that $w_{ii} = 0$.

We define the degree matrix D, which is a diagonal matrix with vertex degrees $d_i = \sum_{j=1}^{n} w_{ij}$

The primary tool for spectral clustering is the graph Laplacian matrices. Let $L = D - W$ denote the unnormalized graph Laplacian matrix.

The eigenvectors of the Laplacian matrix carry significant structural information regarding the connectivity of the graph. And it forms the foundation of spectral graph partitioning. The k-eigenvector is used to partition the graph into k-partition. After computing the k-first eigenvectors of the Laplacian matrix L, we construct V, the matrix with eigenvectors as columns. Finally, using the k-means algorithm, we cluster the rows of V into clusters $C_1, .., C_K$ [147].

3 Proposed Model

One of the main challenges facing Q-routing(subsection 2.1), specifically, and Q-learning, in general, is the curse of dimensionality[148]. This occurs when the size of the solution space expands exponentially as the state/action space grows. This is due to the use of a tabular storage method for routing estimations, which leads to a rapid increase in Q-table size as the network size increases, making Q-routing impractical for large-scale city networks and limiting its applicability in certain fields.

On the other hand, Q-routing has a slow propagation of traffic information between nodes due to its update mechanism. In Q-routing, the new state of a node is only shared with its adjacent nodes when a vehicle is sent to that specific node. Without this, there is no communication between the node and its neighbors. This design results in a network state update delay proportional to the path length (number of hops), as shown in Figure 3.1. Additionally, it can lead to traffic being routed through a broken path that has not yet been updated at the forwarding node.

Our proposed architecture offers two advantages: it reduces unnecessary computation when routing within a local cluster, thus decreasing the size of Q-tables and minimizing hardware requirements. Additionally, it simplifies routing to remote nodes outside of clusters by pre-assigning entry and exit nodes to distant clusters, which reduces computation time for the algorithm.

3.1 Agents definition

The proposed architecture includes three distinct and autonomous agent types, each with a specific set of tasks as shown in Figure 3.3.

❑ **Vehicle Agent(VA) :** The proposed architecture includes vehicle-installed agents that

handle communication (request/response) between the vehicle and the "routing agents". Their primary purpose is to request a route from "routing agents" whenever a vehicle approaches an intersection, providing its desired destination as input.

- ❑ **Inner Cluster Routing Agents (ICRA) :** The proposed architecture also includes node-installed agents that have two main responsibilities:

  1. To provide local routing to destinations within the inner cluster using its local routing state estimation as per the Equation 3.2.
  2. If the destination is outside the local cluster, it provides the route to the local cluster's best exit node.

- ❑ **Outer Cluster Routing Agent (OCRA):** The proposed architecture also includes cluster exit/entry node-installed agents which are responsible for routing to destinations outside the local cluster by routing to the best entry node for the destination's cluster.

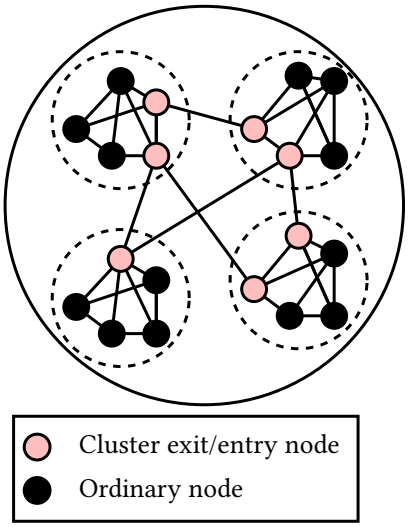


Figure 3.3: Traffic network clustering.

3.2 Routing algorithm

The routing algorithm process, as described in Figure 3.4 and Algorithm 1, allows for multiple routing agent types to be installed in the same node depending on the node's position in the network topology.

Algorithm 1 starts with a check if the vehicle is at an exit/entry node. If it is, then it routes the vehicle to the destination's cluster. If the routing is for outer routing, then the vehicle is routed to the cluster exit node. And if the routing is for inner routing, then the vehicle is routed to the destination node d using Equation 3.2. The algorithm terminates when the vehicle reaches the destination node d.

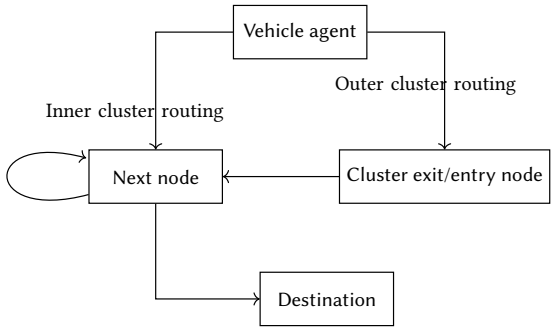


Figure 3.4: The routing process diagram.

Algorithm 1 Routing algorithm

Ensure: Route vehicle v from a node s to a node d

while not at destination d **do**
  if v at exit/entry node **then**
    route v to destination's cluster
  end if
  if Outer routing **then**
    route v to cluster exit node
  end if
  if Inner routing **then**
    route v to destination d
  end if
end while

4 Results and Discussion

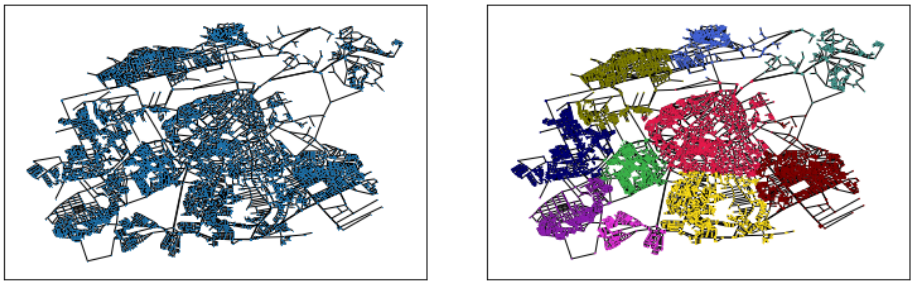
The proposed architecture has been implemented and tested on the real-world network of the city of Fez, Morocco, which has a surface area of $96.5km^2$. Fez is known for being one of the most densely populated and crowded cities in Morocco.

The network used for testing is comprised of $n = 11902$ nodes and $m = 18167$ edges and has an average shortest path length of $a = 56.927$. The city network Figure 3.5a was obtained from OpenStreetMap [149] and was divided into 10 clusters using spectral clustering as previously described.

The clustering results, as shown in Figure 3.5b, demonstrate that the network was effectively divided into clusters and that the clusters are separated by a small number of nodes that serve as entry/exit points between different homogeneous clusters.

The implementation and testing on a real-size network in Fez, Morocco showcases the effectiveness of the proposed architecture in handling real-world scenarios and its ability to adapt to different network topologies and sizes. The number of clusters can also be adjusted depending on the specific requirements and characteristics of the city network being analyzed.

After achieving an adequate clustering of the network, multiple simulated routing scenarios were conducted to compare the proposed architecture with traditional routing



(a) (b)

Figure 3.5: The city of Fez, Morocco. (a) before clustering and (b) after clustering into 10 distinct clusters.

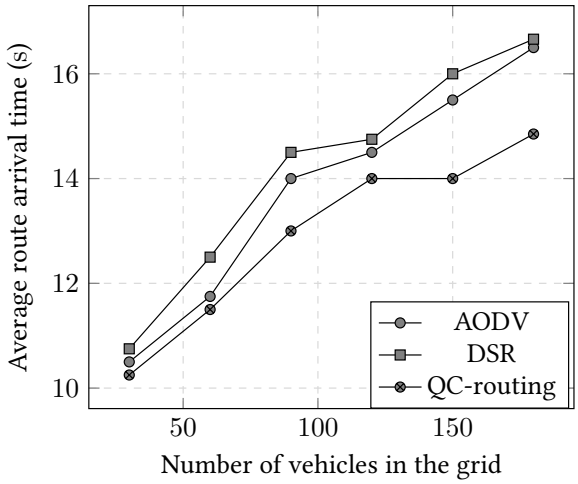


Figure 3.6: Average vehicle arrival time under normal traffic conditions.

algorithms (AODV) and (DSR) in normal traffic conditions, as shown in Figure 3.6. Additionally, the algorithms were also compared in scenarios where congestion was dynamically added and random network state changes were introduced to test the algorithm's ability to handle dynamic changes to the network, as shown in Figure 3.7. The congestion was simulated by randomly blocking certain routes for a period of time, requiring reroutes to be performed.

In both scenarios, the overall travel time of vehicles inside the network was studied and it was found that the proposed algorithm outperformed (AODV) and (DSR) by providing better travel time for the overall network. The better results can be attributed to the proposed algorithm's ability to predict and adjust routes before the complete shutdown of a route occurs. These results, given the stochastic nature, demonstrate the proposed algorithm's suitability for real-world scenarios.

Furthermore, it was also observed that as the number of clusters increased, there was a significant decrease in table size, indicating efficient usage of system memory, as shown in Figure 3.8.

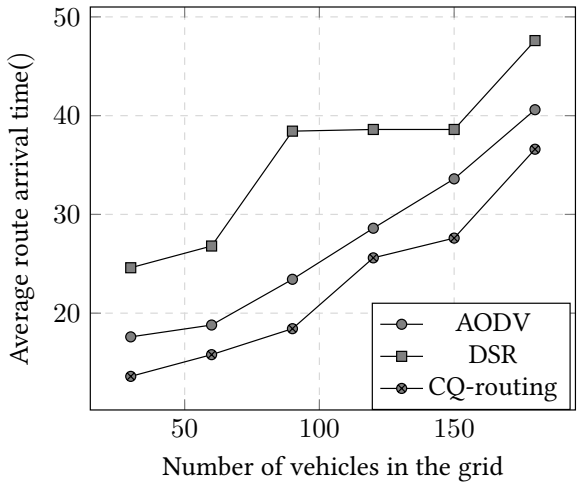


Figure 3.7: Average vehicle arrival time under stochastically generated traffic congestion

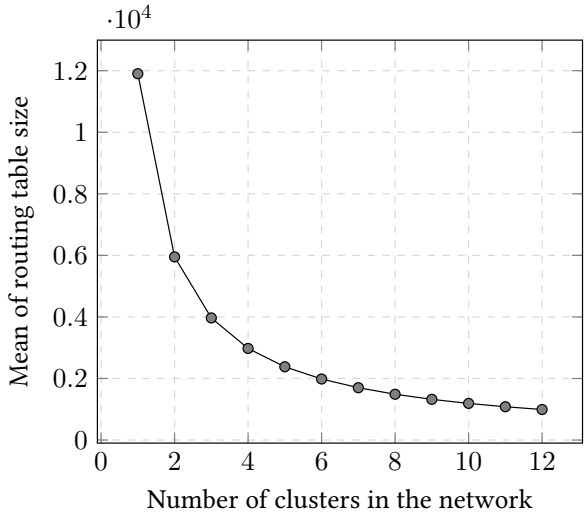


Figure 3.8: Mean routing table size.

5 Conclusion

The growing traffic and congestion in cities have made it crucial to develop efficient routing solutions that can handle large networks. However, most current models struggle in real-world traffic scenarios due to their inability to adapt to dynamic and frequent changes in network state or unexpected route blockages in the network topology.

In this new approach, we proposed an architecture that provides an optimal routing solution in highly dense urban areas by partitioning the network into multiple districts using spectral clustering, intending to minimize the number of connections between the districts. Additionally, we separated the routing policy into two phases: internal cluster routing and external clustering routing using Q-routing. This approach significantly reduces the

hardware resources and calculation time required because it reduces the action space.

The partitioning of the network into multiple clusters creates multiple homogeneous areas with concentrated traffic, minimizing the need to compare routes with distant nodes.

We tested the proposed algorithm through simulations based on two scenarios: one with normal traffic conditions and the other with stochastic congestion to evaluate the system's ability to adapt to the random nature of vehicle traffic. The results showed that the proposed algorithm outperformed other algorithms (AODV and DSR) and adapted well to changes in traffic congestion levels.

CHAPTER 4

The problem of Vehicle overtaking

1 Introduction

In recent years, there have been significant advancements in traffic safety features, which have provided newer vehicles with important safety properties. One such feature is the inclusion of airbags, which help to reduce injuries during collisions by absorbing the impact of the crash within the vehicle, such as the steering wheel. Lane departure warning systems alert the driver if the vehicle starts to drift off the center line of the lane, helping to prevent accidents caused by inattention or fatigue. Over-speed warning devices, on the other hand, alert the driver when they exceed the allowed speed limit for a given road segment, helping to reduce the likelihood of accidents caused by reckless or careless driving. Other safety features that have been developed include anti-lock braking systems and brake assist systems, which provide better braking in emergencies, and automatic emergency braking, obstacle detection, and left turn crash avoidance systems, which use sensors to detect other vehicles and automatically activate the brakes in order to avoid accidents. All of these features work together to improve the overall safety of vehicles on the road.

More recent safety features in vehicles include automatic emergency braking, obstacle detection, and left-turn crash avoidance systems. These systems use sensors to detect other vehicles and automatically activate the brakes in order to prevent collisions. Driver drowsiness detection systems are also an important safety feature that has been developed in recent years. These systems use various indicators, such as steering patterns or face monitoring, to detect a driver's level of sleepiness and alert them if they are at risk of falling asleep at the wheel. These systems are particularly useful for truck drivers who often take long routes and are at risk of accidents caused by drowsiness. Overall, these safety features have greatly enhanced the safety of vehicles and have the potential to prevent a significant number of accidents.

However, one of the remaining challenges in the driving task, faced by both autonomous systems and human drivers, is the overtaking maneuver, which has not seen much progress in academic research and is therefore unlikely to be used in consumer vehicles anytime soon. The lack of progress in this area can be attributed to several factors, which can be broadly divided into perception-related issues (e.g., detection) and algorithmic issues (e.g., planning). Perception-related issues refer to the difficulty of accurately perceiving the

environment, which can be done using a variety of sensors such as LiDARs, cameras, radar, GPS, etc. However, the task of processing and interpreting this information is challenging, depending on the system's objectives. Planning-related problems are even more difficult to solve and require significant research effort to develop optimal routing, navigation, and control algorithms, as well as testing and optimization efforts. It's worth noting that most of these algorithms require the use of state-of-the-art vehicles equipped with expensive hardware.

When attempting an overtaking maneuver, safety must be the top priority, taking into account various factors such as the presence of approaching vehicles from both front and rear, the relative speed and distance of the surrounding vehicles, the number of leading vehicles, and the gap between the leading and host (ego) vehicles. These factors can be challenging for human drivers to accurately assess, making it unrealistic to rely solely on their judgment to evaluate the safety of executing this action.

To aid drivers in assessing safety during overtaking, traffic signals such as "No passing" signs and road markings are frequently utilized in actual driving situations. However, in many underdeveloped and developing countries, these signals are either missing or inaccurate. Additionally, on rural roads, there is a lack of compliance with traffic regulations, resulting in drivers taking dangerous risks while changing lanes, especially on winding roads.

The challenges of achieving safe overtaking are numerous, particularly when it comes to the reliability of assessing the safety of a maneuver before it is initiated. This is due to the dynamic and random nature of traffic and driver behavior, particularly on two-lane roads with oncoming traffic. Overtaking at high speeds and acceleration makes it almost impossible to abort faulty maneuvers. Additionally, the system must process sensor data in real-time, so it must be precise to avoid any errors that may compound.

Overtaking requires maneuvering a vehicle around another vehicle while avoiding accidents, which requires both horizontal and longitudinal movements. It can be challenging due to various environmental factors such as road rules, visibility, weather, and the presence of different types of vehicles. The process of overtaking typically consists of three sub-tasks: moving to the overtaking lane, passing the slower vehicle, and returning to the original lane. The lane change sub-task, which marks the beginning and end of an overtake, can be classified as either optional or required.

Accurately assessing the safety of initiating an overtake maneuver is one of the main difficulties in completing it successfully. This is because several factors come into play, such as the presence of oncoming vehicles from the front or rear, the relative speed and distance of surrounding vehicles, the number of leading vehicles, and the gap between the leading and host vehicles. These factors can be challenging for a human driver to estimate, making it unrealistic to rely solely on their ability to assess the safety of an overtake maneuver. To address this challenge, an architecture is proposed that involves two primary stages: object detection and tracking, and vehicle control during the overtake maneuver.

The initial stage, object detection and tracking involve utilizing sensors such as LiDARs, cameras, and radar to accurately perceive and interpret the surrounding environment. This stage is crucial in identifying potential hazards that may affect the safety of the overtaking maneuver. The second stage, controlling the vehicle during the overtaking maneuver, involves using algorithms to plan and execute the overtake safely and efficiently. This

stage requires substantial research to develop optimal routing, navigation, and control algorithms, as well as testing and optimization to ensure their reliability. By addressing both object detection and tracking and vehicle control during the overtaking maneuver, it is believed that it will be possible to significantly improve the safety and reliability of this important driving task.

2 Overtaking strategies

Vehicle overtaking can be divided into five different tactics:

- **Accelerative:** when a faster vehicle approaches a slower vehicle, the following vehicle will slow down and match the speed of the leading vehicle for a certain amount of time, depending on the gap available in the opposing lane. Once the following vehicle finds enough space in the opposing lane, it will change lanes and accelerate to its maximum speed to complete the overtaking maneuver.

- **Flying:** when a faster vehicle approaches a slower vehicle in a free-flowing traffic stream and there are no vehicles nearby, the following vehicle (faster vehicle) will change lanes from a long distance without slowing down.

- **Piggybacking:** when an overtaking vehicle follows another vehicle that is overtaking a slower vehicle. For example, a vehicle in front overtakes the preceding vehicle and the subject vehicle follows this vehicle to perform the maneuver, both vehicles overtake the preceding vehicle in the same maneuver. Acceleration is still required and the maneuver is completed by passing the preceding vehicle.

- **Two plus (2+):** a vehicle performs an accelerative maneuver and takes over two or more vehicles in a single action.

In this research we will be only focusing on Accelerative overtaking, however, both the piggybacking and the two plus strategies can be modeled based on the accelerative approach

2.1 The overtaking maneuver phases

Overtaking is a frequent occurrence on the road and is often driven by the speed difference between the main vehicle and other vehicles, as well as each driver's desire to maintain their desired speed while driving. It is a common practice for drivers and is typically done to accommodate for speed differences between vehicles on the road and to meet the specific needs of drivers to maintain their preferred speed. Additionally, overtaking is often necessary for reducing traffic congestion and avoiding long lines of vehicles.

A two-lane highway overtaking maneuver can be broken down into three Figure 4.1:

Stage 1: starts at the beginning of the overtaking action and ends when the vehicle has fully transitioned to the opposite lane and its front is aligned with the vehicle in front.

Stage 2: starts at the end of stage 1, and it's part of the maneuver where the vehicle passes the leading car and travels straight ahead on the opposite lane. At the end of this stage, the vehicle's front and rear are aligned.

Stage 3: begins after stage 2 and continues until the vehicle completes the overtaking move by returning to its original lane.

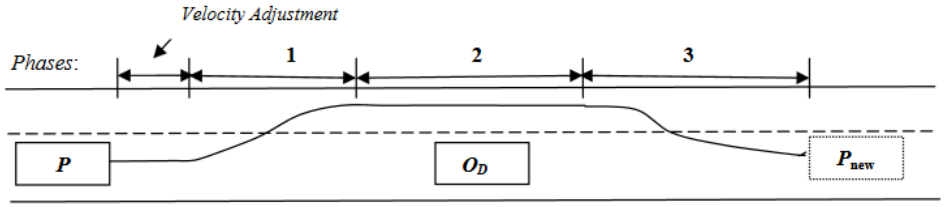


Figure 4.1: The Overtaking maneuver.

3 Vehicle modeling

Choosing an effective and reliable control architecture for vehicles involves selecting a system model that balances accuracy and computational efficiency. In the realm of autonomous vehicles, there are several models available, each with its strengths and weaknesses depending on the specific task. These models include linear point mass, non-linear point mass, kinematic or dynamic bicycle, and full four-wheel models. To accurately represent the vehicle's behavior, it is important to choose the appropriate control variables based on environmental and maneuver constraints. However, using an overly complex model may have a negative impact on the real-time performance of the control algorithm.

3.1 Frames of Reference

In this study, three different reference frames will be used to describe the position and orientation of the vehicle in space. These frames are:

- The fixed inertial frame is associated with the ground, denoted by the XY-axis.
- The body-fixed frame associated with the vehicle with the origin at the Center of Gravity (COG), with the x-axis and y-axis representing the longitudinal and lateral directions, respectively.
- The third local frame of reference is the front wheel axis xw-yw.

3.2 Kinematic Bicycle Model

The Kinematic Bicycle Model is one of the simplest models used to represent the motion of a vehicle. As the name suggests, it describes the motion of a system such as a point or body without considering the forces and torques that caused it. Therefore, it only describes

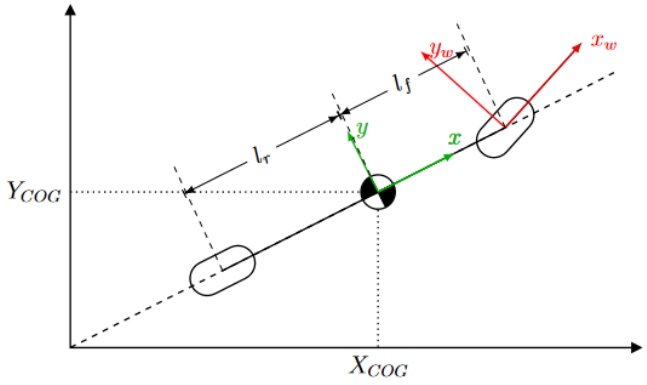


Figure 4.2: Reference Coordinate Frames

the geometry of the motion of the system. In this model, the bicycle is represented by a single front wheel and a single rear wheel. It also assumes that the inner and outer wheel steering angles are equal and that the steer angles and tire slip angles for the left and right tires are combined into effective values representing the two wheels (Figure 4.3).

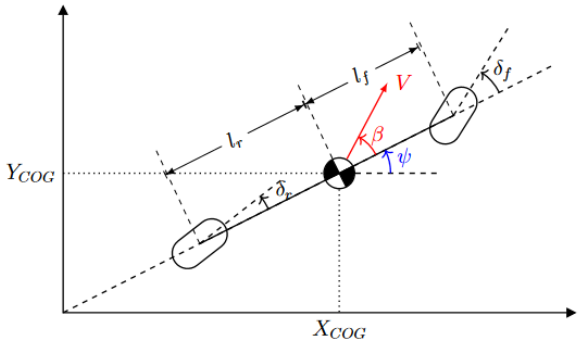


Figure 4.3: Kinematic Bicycle Model.

The kinematic equations of motion for the Kinematic Bicycle Model are derived as follows:

$$\dot{X} = V\cos(\beta + \psi) \tag{4.1a}$$

$$\dot{Y} = V\sin(\beta + \psi) \tag{4.1b}$$

$$\dot{\psi}(t) = \frac{V\cos(\beta)}{l_f + l_r}(\tan(\delta_f) - tan(\delta_r)) \tag{4.1c}$$

$$\beta = tan^{-1}(\frac{l_f\tan(\delta_r) + l_r\tan(\delta_f)}{l_f + l_r}) \tag{4.1d}$$

For front-wheel steering vehicles, we assume $\delta_r = 0$, Therefore:

$$\dot{\psi}(t) = \frac{V\cos(\beta)\tan(\delta_f)}{l_f + l_r} \tag{4.2a}$$

$$\beta = tan^{-1}(\frac{l_r\tan(\delta_f)}{l_f + l_r}) \tag{4.2b}$$

The kinematic model assumes that the slip angle is zero, implying that the velocity vectors of the front and rear wheels are in the same direction as their corresponding wheels. This assumption is valid for low lateral accelerations, which are commonly observed at low speeds. This is a simplifying assumption that facilitates a more manageable mathematical model, however, it does not consider the effects of tire slip, which can affect the vehicle's behavior at higher speeds or on slippery surfaces.

3.3 Dynamic Bicycle Model

In contrast to the kinematic model, the dynamic model utilizes forces to execute movements. The kinematic and dynamic models have several distinctions. The dynamic model is more suitable for vehicles executing high-speed maneuvers because it does not assume negligible slip angles on the front and rear wheels, as seen in the dynamic bicycle model (Figure 4.4).

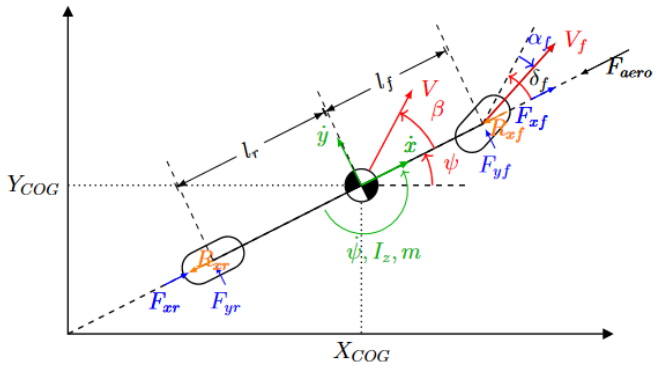


Figure 4.4: Dynamic Bicycle Model.

Figure 4.5 illustrates the different forces acting on the vehicle's body-fixed frame, including the longitudinal tire force F_x, the lateral tire force F_y, the tire-fixed frame forces (F_{l*} and F_{c*}), rotational and translational velocities, and the front wheel steering angle δ.

$$\ddot{x} = \frac{1}{m}(F_{xr} + F_{xf} + m\dot{\psi}\dot{y}) \tag{4.3a}$$

$$\ddot{y} = \frac{1}{m}(F_{yr} + F_{yf} - m\dot{\psi}\dot{x}) \tag{4.3b}$$

$$\ddot{\psi} = \frac{1}{I_z}(l_f F_{yf} - l_r F_{yr}) \tag{4.3c}$$

The yaw angle, also known as the heading angle, of a vehicle in the inertial frame, is represented by $\dot{\psi}$. This refers to the angular velocity of the vehicle with respect to the

Table 4.1: Notations.

Symbol	Name
F_{xf},F_{yf}	front tire forces
F_{xr},F_{yr}	rear tire forces
$\dot{x}$,$\dot{y}$	velocity coordinates in the body-fixed frame
ψ	heading angle (yaw), in the inertial frame
I_z	moment of inertia about the z-axis
δ	front wheels steering angle
α	slip angle
ρ	slip ratio
C_α	cornering stiffness for the front and rear tires
C_ρ	longitudinal stiffness
r	tire radius
ω	angular velocity of the tire
l_f, l_r	distance from the COM point to the front and rear tires respectively

global axis, measured at the center of gravity (COG). The forces exerted by the tires are expressed in the tire-fixed coordinate frame. When the vehicle is moving in a straight line and accelerating or braking, the longitudinal tire forces (F_{xf} and F_{xr}) are generated. When the vehicle is turning, the lateral tire forces (F_{yf}, F_{yr}) are generated. The vehicle's global position is defined as:

$$\dot{X} = \dot{x}cos(\psi) - \dot{y}sin(\psi) \tag{4.4a}$$
$$\dot{Y} = \dot{x}sin(\psi) + \dot{y}cos(\psi) \tag{4.4b}$$

The motion of the vehicle can be controlled by tracking a trajectory that is generated by following the center line of the leftmost lane, as illustrated in Equation 4.3.

3.4 Tire forces

Calculating tire forces is crucial to understanding the mechanics of a vehicle, particularly when it is moving at high speeds. High-speed maneuvers, such as overtaking, are typically performed on paved roads, where both slip angle α and slip ratio ρ are small. As a result, a linear tire model can be utilized to determine lateral tire forces, which can be represented using a linear tire model equation.

$$F_{yf} = C_\alpha \alpha_f \tag{4.5a}$$
$$F_{yr} = C_\alpha \alpha_r \tag{4.5b}$$
$$F_{xf} = C_\rho \rho_f \tag{4.5c}$$
$$F_{xr} = C_\rho \rho_r \tag{4.5d}$$

The cornering stiffness coefficient C_α for the front and rear tires and the longitudinal stiffness C_ρ can be defined as follows:

$$\alpha_f = \frac{\dot{y} + l_f \dot{\psi}}{\dot{x}} - \delta \tag{4.6a}$$

$$\alpha_r = \frac{\dot{y} - l_r \dot{\psi}}{\dot{x}} \tag{4.6b}$$

$$\rho_* = \begin{cases} \frac{rw - \omega_*}{r\omega} - 1 & \text{during braking} \\ 1 - \frac{r\omega}{r\omega - \omega_*} & \text{during acceleration} \end{cases} \tag{4.6c}$$

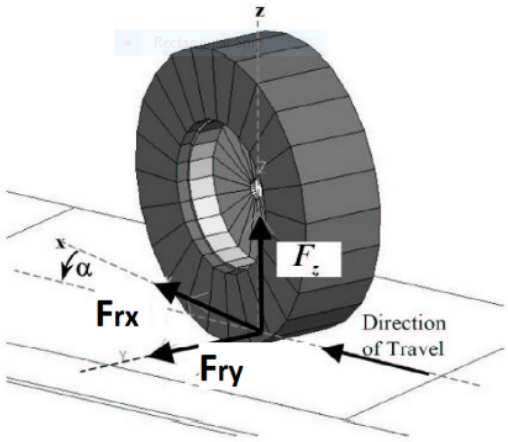


Figure 4.5: A scheme of the tire forces, notations are for the rear wheel.

3.5 Spacial dynamics vehicle model.

The spatial coordinate pair (e_ψ, e_y) represents the center of gravity (COG) of the vehicle. The parameter e_y indicates the deviation of the vehicle from its lateral position on a curved path, while e_ψ denotes the deviation from the path's centerline, which is represented by ρ. The tire's radius is denoted as ϕ_s, and the vehicle's position along the centerline of the road is denoted as s. This coordinate transformation allows for the calculation of the vehicle's dynamics along the centerline of a lane, as illustrated in the curvilinear coordinate system shown in Figure 4.6.

$$\dot{s} = \frac{\rho}{\rho - e_y}(\dot{x}\cos(e_\phi) - \dot{y}\sin(e_\phi)) \tag{4.7}$$

The derivative with respect to s:

$$e'_y = (\dot{x}\sin e_\psi + \dot{y}\cos e_\psi)/\dot{s} \tag{4.8a}$$

$$e'_\psi = \dot{\psi}/\dot{s} - \psi'_s \tag{4.8b}$$

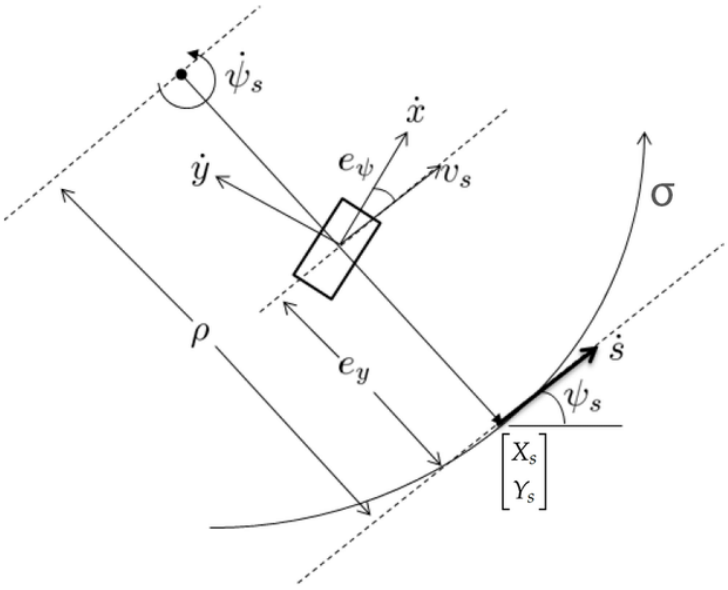


Figure 4.6: Curvilinear coordinate system [150].

4 Object detection and tracking

4.1 You Only Look Once (YOLO)

You only look once (YOLO) is a convolutional detection network used for real-time object detection. It was first introduced in [151] and is popular for fast and precise object detection scenarios where it can detect multiple objects from different classes in the same image.

Unlike Region Based Convolutional Neural Networks (R-CNN), YOLO views object detection as a single regression problem and feeds the entire image to the neural network, making it faster but with lower accuracy. The image is divided into SxS cells, where each cell is responsible for detecting objects whose center points fall within the grid. Each cell predicts a tuple of 4 values (center x-coord, center y-coord, width, height) that define the bounding box, which is normalized by the image width and height. Additionally, a confidence score indicating the probability that the cell contains an object and the probability of the object belonging to each class are predicted if the cell contains an object. Duplicate bounding boxes are eliminated by Non-Maximum Suppression after detection, as shown in (Figure 4.7).

The architecture of the network includes initial layers that use convolution to extract information from the image, while the fully connected layers are responsible for determining the output probabilities and coordinates. The architecture consists of 24 convolutional layers, followed by 2 fully connected layers. (as depicted in Figure 4.8)

A research paper by [152] presented an improved version of the YOLO algorithm known as YOLOv2. This updated algorithm introduced several new features, including a high-resolution classifier, batch normalization, the use of anchor boxes in convolution, dimension clustering, and fine-grained features to enhance object detection capabilities. Subsequently, in another paper published in [153], additional advancements were proposed

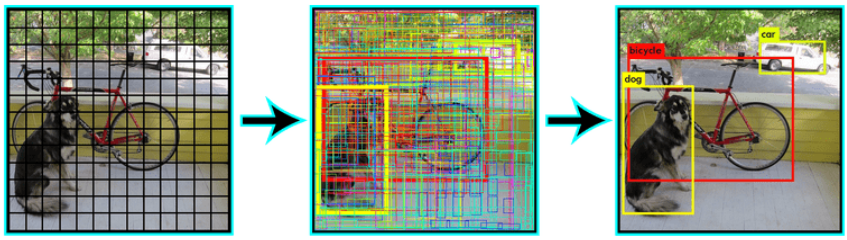


Figure 4.7: YOLO for object detection.

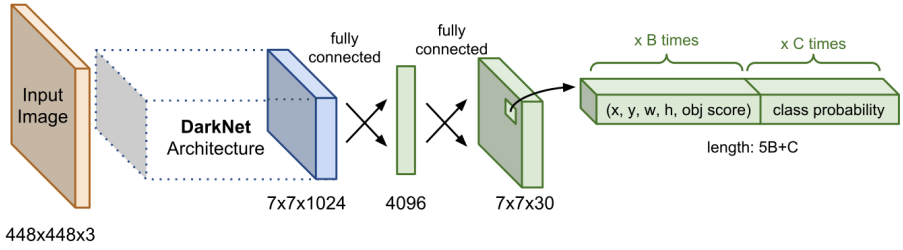


Figure 4.8: YOLO network architecture.

to further enhance the accuracy and speed of the network. These improvements included multiscale detection, multi-label classification, and the integration of a more advanced feature extraction network called Darknet-53 into the YOLO architecture.

YOLOv3 offers a balance of high accuracy and efficient speed by incorporating various state-of-the-art technologies such as a residual block-based backbone, feature pyramid network-inspired network head for multi-scale detection, batch normalization and anchor box prediction. The feature extractor used in YOLOv3 is called Darknet-53, it is composed of 53 convolutional layers without max-pooling. It performs convolution, followed by Batch Normalization, and leaky RELU for each convolution operation. YOLOv3, released in 2018, built upon the capabilities of YOLOv2 by introducing a new architecture known as Darknet-53, which is comprised of 53 convolutional layers. It also added support for object tracking and improved accuracy while decreasing computation time through the use of anchor boxes. Additionally, it employed a method referred to as a Feature pyramid network (FPN) that combines features from different layers to enhance the detection of small objects.

Darknet-53 is a powerful backbone for extracting features from images (Figure 4.9). It is comprised of 53 convolutional layers and advanced structures such as (a) Residual blocks which provide shortcut layers to ease the training process of the network; (b) Inception structure containing 3x3, 1x1 convolutional kernels which maintain the field of view while reducing computation cost; (c) Batch normalization layer which enhances the independence of each layer in the network during the learning process.

The YOLOv4 algorithm, as described in [154], features improved feature extractors through the use of CSPDarknet53 (Cross-Stage-Partial connections). This enhancement has led to significant improvements in both the speed and accuracy of the algorithm. The YOLOv4 network generates B-bounding boxes and calculates the confidence score for

Layer	Filters size	Repeat	Output size
Image			416 × 416
Conv	32 3 × 3/1	1	416 × 416
Conv	64 3 × 3/2	1	208 × 208
Conv Conv Residual	32 1 × 1/1 64 3 × 3/1	Conv Conv Residual × 1	208 × 208 208 × 208 208 × 208
Conv	128 3 × 3/2	1	104 × 104
Conv Conv Residual	64 1 × 1/1 128 3 × 3/1	Conv Conv Residual × 2	104 × 104 104 × 104 104 × 104
Conv	256 3 × 3/2	1	52 × 52
Conv Conv Residual	128 1 × 1/1 256 3 × 3/1	Conv Conv Residual × 8	52 × 52 52 × 52 52 × 52
Conv	512 3 × 3/2	1	26 × 26
Conv Conv Residual	256 1 × 1/1 512 3 × 3/1	Conv Conv Residual × 8	26 × 26 26 × 26 26 × 26
Conv	1024 3 × 3/2	1	13 × 13
Conv Conv Residual	512 1 × 1/1 1024 3 × 3/1	Conv Conv Residual × 4	13 × 13 13 × 13 13 × 13

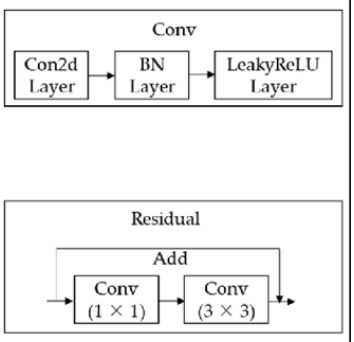


Figure 4.9: Structure of the Darknet53 convolutional network.

each box, where the bounding boxes indicate the position of the detected objects (i.e., their coordinates, width, and height) relative to the input image, and the confidence score represents the model's level of certainty that the box contains an object.

The YOLOv4 architecture is composed of four main blocks: the backbone, the neck, the dense prediction, and the sparse prediction, as illustrated in the image (Figure 4.10). The backbone is the feature extraction architecture, which is CSPDarknet53, it splits the current layer into two parts, one that goes through convolution layers, and the other that doesn't, then the results are combined.

The neck is used to add layers between the backbone and the dense prediction block (head). YOLOv4 architecture uses a modified Path Aggregation Network, a modified Spatial Attention Module, and a modified Spatial Pyramid Pooling, which are all used to aggregate the information and improve accuracy.

The head is used for locating bounding boxes and for classification, similar to YOLOv3. The coordinates of the bounding box (x, y, height, and width) are detected, as well as the score. The output is a vector with the bounding box coordinates and the probabilities of the classes.

The YOLOv4 algorithm employs additional techniques to improve accuracy during training and inference, called "Bag of Freebies" and "Bag of Specials".

- The Bag of Freebies, which is used during training, improves accuracy without increasing inference time. It has two techniques:
  - Bag of freebies for the backbone, which uses Cut Mix and Mosaic for data augmentation, and Drop Block for regularization.
  - Bag of freebies for detection, which adds more to the backbone, such as Self-Adversarial Training and Random Training Shapes.

- The Bag of Specials changes the architecture and increases the inference time slightly. It also has two techniques:
  - Bag of specials for the backbone, which uses Mish activation and Cross-Stage Partial connections.
  - Bag of specials for detection, which uses SPP-block and Segment Anything Model (SAM) block.

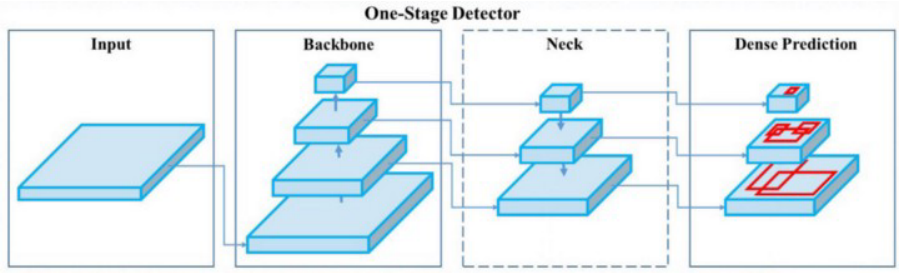


Figure 4.10: Yolov4 Object detector.

Released in 2021, YOLOv5 is the latest version of the YOLO object detection system, which offers improvements over YOLOv4. PyTorch is used in YOLOv5 instead of Darknet, which enhances support and development while delivering better performance. This version boasts a deep neural network with 283 layers and over 7 million trainable parameters. To extract critical features from input images, YOLOv5 employs Cross Stage Partial Networks (CSP) as its backbone. The neck of YOLOv5 creates feature pyramids, while the head predicts targets of varying sizes on feature maps.

Additionally, Simple Online and Realtime Tracking (DeepSORT) can be used for object tracking across multiple frames after object detection is completed. DeepSORT is an online tracking algorithm that employs information only from the current and previous frames to make predictions about the current frame, without the need to process the entire video.

In Computer vision, two types of object detection algorithms are currently available based on deep learning: one-stage and two-stage.

❑ **One-stage Object Detection:** Algorithms such as YOLO, Single Shot Detector (SSD), SqueezeDet, and DetectNet, are often used when speed and efficiency are required, like on embedded systems and mobile devices. These algorithms require only a single pass through the neural network to predict all the bounding boxes and the class confidence in one go.

❑ **Two-stage Object Detection:** Algorithms such as the R-CNN series, start by generating the bounding box proposal using a Region proposal network (RPN). Then, the proposed region is classified using Convolutional neural network (CNN). These two-stage object detection algorithms have the advantage of higher accuracy than the one-stage algorithms but are much slower in comparison.

4.2 Object depth estimation

A reliable method for monitoring the surrounding environment is essential in evaluating overtaking safety in a vehicle. One possible solution is a system that utilizes two front-mounted cameras to provide stereoscopic vision mapping, allowing for the identification of both stationary and moving objects on the road. Additionally, by capturing multiple consecutive images, the vehicle's relative speed can be determined.

Stereoscopic vision is a well-established technique for calculating the depth of objects, which involves two horizontally aligned cameras that are separated vertically by a distance A. This technique is based on the same principles that enable humans to perceive depth in their field of vision. By utilizing the known properties of the cameras and geometric principles, stereoscopic vision can accurately calculate the distance between the cameras and an object.

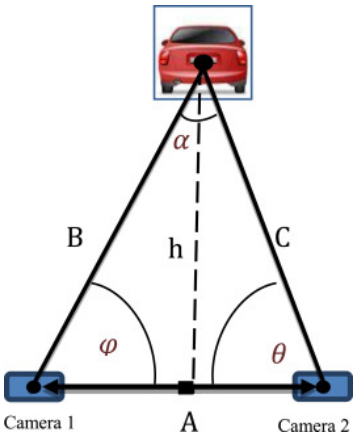


Figure 4.11: An illustration of two cameras mounted together as a stereo camera.

In summary, the process of stereoscopic vision includes four main steps. The initial step is capturing two synchronized images from the two cameras. The next step is processing these images, which includes rectifying the images to align them and then performing image matching to find corresponding pixels in both images. The third step is object detection and tracking, this helps identify and track the objects of interest in the images. The final step is using geometric triangulation to calculate the 3D coordinates of each object. This is done by utilizing the known properties of the cameras and the disparities between corresponding pixels in the two images. Once the 3D coordinates of the objects are known, it is possible to calculate the relative velocity of the objects by using the distance between the objects in multiple consecutive frames [155].

As shown in Figure 4.11, the two cameras are horizontally mounted and separated by a distance A. where h is the desired distance to estimate between the object and the cameras. Using trigonometric functions, the distance between the object and the left camera is represented by B, and the distance between the object and the right camera is represented by C. $\theta, \phi and \alpha$ are the angles of the triangle formed by the object and the two cameras.

$$\sin(\theta) = \frac{h}{C} \tag{4.9a}$$

$$\sin(\varphi) = \frac{h}{B} \tag{4.9b}$$

Therefore,

$$h = B\sin(\varphi) = C\sin(\theta) \tag{4.10a}$$

Using the law of sins :

$$\frac{A}{\sin(\alpha)} = \frac{B}{\sin(\theta)} \tag{4.11a}$$

So:

$$B = \frac{A\sin(\theta)}{\sin(\alpha)} \tag{4.12a}$$

$$h = \frac{A\sin(\theta)\sin(\varphi)}{\sin(\alpha)} \tag{4.13a}$$

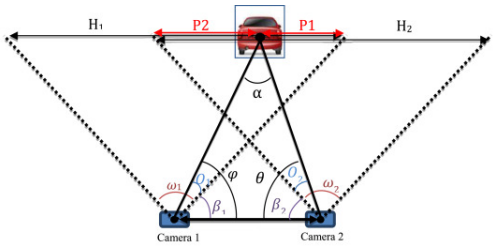


Figure 4.12: Illustration of the angles used for computing the distance.

According to Figure 4.12:

w_1, w_2: the view angles of the two cameras respectively. $H1, H2$: the number of horizontal pixels of the two cameras respectively. $P1, P2$: the position of the object in both cameras, where $P1$ is the distance in pixel between the centroid of the object and the end of the overlap area for the camera on the left.$P2$ is the distance in pixel between the centroid of the object and the beginning of the overlap area for the right camera.

Therefore,

$$\phi = O_1 + \beta_1 \tag{4.14a}$$

$$\theta = O_2 + \beta_2 \tag{4.14b}$$

$$\beta_1 = \frac{180 - w_1}{2} \tag{4.14c}$$

$$\beta_2 = \frac{180 - w_2}{2} \tag{4.14d}$$

Now β_1 and β_2 are known, so we still need $O1$ and $O2$. These two angles can be obtained by multiplying the position of the object in both cameras (P1 and P2) by angles that correspond to each pixel in the two cameras (Ap1 and Ap2), as shown below:

$$O_1 = P1Ap1 \tag{4.15a}$$
$$O_2 = P2Ap2 \tag{4.15b}$$

AP1 and AP2 can be calculated using :

$$Ap1 = \frac{w_1}{H_1} \tag{4.16a}$$
$$Ap2 = \frac{w_2}{H_2} \tag{4.16b}$$

Therefore :

$$\phi = P1\frac{w_1}{H_1} + \beta_1 \tag{4.17a}$$
$$\phi = P2\frac{w_2}{H_2} + \beta_2 \tag{4.17b}$$
$$\alpha = 180 - (\phi + \theta) \tag{4.17c}$$

Finally :

$$h = \frac{A\sin(P2\frac{w_2}{H_2} + \beta_2)\sin(P1\frac{w_1}{H_1} + \beta_1)}{\sin(180 - P1\frac{w_1}{H_1} + \beta_1 + P2\frac{w_2}{H_2} + \beta_2)} \tag{4.18a}$$

5 Model predictive control

As technology advances, the importance of control strategies in industrial applications has grown exponentially. Specifically, systems with multiple constraints that must be managed by a single controller, such as Multi-input multi-output (MIMO) systems, have become increasingly prevalent. One of the most widely utilized control strategies in these scenarios is Model Predictive Control. The ability of Model predictive control (MPC) to handle constraints, and optimize processes online, and its simple design make it a favorite among practitioners and researchers. In fact, MPC was first introduced in industrial applications as far back as the 1970s.

While there are various predictive controllers available, each with its unique properties, such as Generalized Predictive Control, Dynamic Matrix Control, Unified Predictive Control, Internal Model Control, and Extended Model-based Predictive Control, MPC currently reigns as the most preferred control strategy for process industries. It is widely utilized in fields like petrochemicals and refining, food processing, automotive, metallurgy, aerospace, and defense.

An MPC algorithm consists of a cost function, constraints, and a mathematical model of the system that represents the complex behaviors of dynamic systems. This model is used to predict the system's future state over a future time interval. The ultimate goal of the MPC controller is to limit the violation of input and output constraints and optimize input and output variables to stay within the assigned limits. The structure of the MPC is shown in Figure 4.13 :

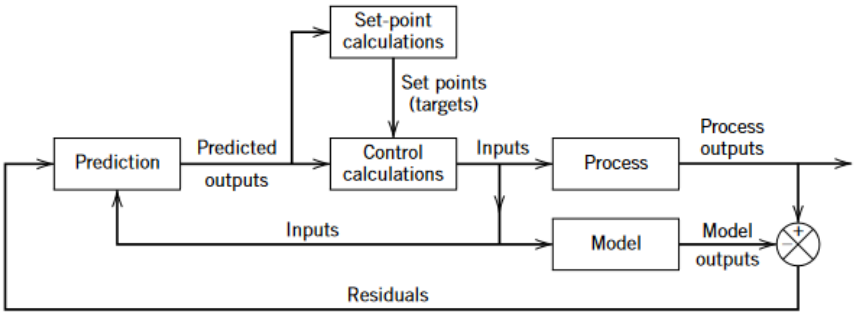


Figure 4.13: Structure of MPC.

The system's output is determined by its current and past values as well as future control actions to calculate optimal control signals for future actions. Model Predictive Control (MPC) is an approach that involves solving an optimal control problem repeatedly for the current system state on a finite horizon H of N sampling instants, based on specific criteria. However, only the first part of the resulting input signal is utilized by the system. At each time increment t, environmental constraints are considered while optimizing a performance index. To ensure accurate predictions and optimized control actions, an appropriate system model that can effectively describe the system's future states based on various inputs over the horizon H is crucial (Figure 4.14).

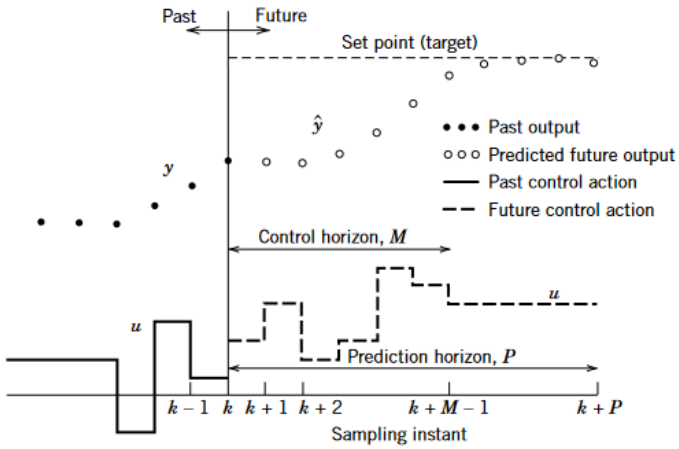


Figure 4.14: Basic concept of MPC.

5.1 Cost function

The J_k cost function sometimes referred to as the objective or optimization function evaluates the performance of a process over a prediction horizon and is minimized in relation to the control vector for the future. It takes the form:

$$J_k = \sum_{i=1}^{L} (y_{k+i} - r_{k+i})^T Q (y_{k+i} - r_{k+i}) + u_{k+i-1}^T P u_{k+i-1} + \Delta u_{k+i-1}^T R \Delta u_{k+i-1} \quad (4.19)$$

Where $Q_i \in R^{m \times m}$, $Q_i \in R^{r \times r}$ and $R_i \in R^{r \times r}$ are the user defined weighting matrices, L is the prediction horizon.

The problem of controlling a system while adhering to the prediction model and process variable constraints is defined as follows:

$$u_{k|L}^* = \arg\min_{u_{k|L}} J_k(u_{k|L}) \quad (4.20)$$

The goal of the cost function is to minimize the difference between the actual output of the process, represented by $yk + 1|L$, and the desired reference output, represented by $rk + 1|L$, while also reducing the amount of control, represented by $uk|L$, needed to achieve this output.

5.2 Constraints

One of the main benefits of MPC is its capability to handle constraints effectively. Some examples of constraints that are commonly used in MPC include limits on the amplitude of the input, limits on the rate of change of the input, and limits on the output.

Input amplitude constraint: It is amplitude constraints on the input signal which can be mathematically written as :

$$u_{k|L}^{\min} < u_{k|L} < u_{k|L}^{\max} \quad (4.21)$$

Input rate constraint: The limitations on the rate of change are stated as input rate constraints. Mathematically it can be written as:

$$\Delta u_{k|L}^{\min} < u_{k|L} < \Delta u_{k|L}^{\max} \quad (4.22)$$

Output constraints: The limitations on the output are defined as output constraints that can be written as :

$$y^{\min} < y_{k+1|L} < u^{\max} \quad (4.23)$$

5.3 Prediction model

MPC relies on a mathematical representation of the system's input-output behavior. The model used for predictions is derived from the system model, which illustrates the connection between future outputs and future control inputs. A linear dynamic process model can be represented as:

$$y_{k+1|L} = p_L + F_L u_{k|L} \tag{4.24}$$

Where F_L is a constant matrix derived from the system model,p_L is a vector that depends on model parameters and several inputs and outputs that are older than time k.

Although numerous MPC-based approaches have been suggested in the literature, they are frequently evaluated only in 2D simulation environments. This testing approach fails to accurately represent real-world scenarios and provides an insufficient assessment of the practicality of implementing the proposed approach on actual vehicles. Furthermore, most systems involve significant simplifications of the model and make assumptions about the environment, restricting their usefulness. In this study, we introduce an MPC controller that is formulated as a finite-dimensional optimization problem. We accomplish this by discretizing the vehicle dynamics ξ with a sampling time T_s.

$$\xi(k+1) = f(\xi(k), u(k)) \tag{4.25a}$$
$$u(k+1) = u(k) + \Delta u(k) \tag{4.25b}$$

Where the state of the system vector at time k is denoted as $\xi(k)$, while the vector of manipulated variables that the controller will establish is denoted as $u(k)$.

$$\xi = [\dot{x}, \dot{y}, \psi, \dot{\psi}, e_\psi, e_y] \tag{4.26}$$

The control and state sequences must be within the specified sets, with $u(k)$ being in the set of admissible control inputs $\mathbb{U}$ (such as the steering angle, braking torque, and throttle) and $\xi(k)$ is in the set of admissible states for the system $\mathbb{X}$. The dynamics of the system are described by $f(\xi(k), u(k))$. The problem of determining the optimal control inputs for a finite period of time in the future, at time instant k, using an open-loop approach is formulated as follows:

$$\min_{u} \quad J(\boldsymbol{\xi}, \boldsymbol{u}) \tag{4.27a}$$
$$\text{s.t.} \quad \xi(k+1) = f(\xi(k), u(k)), k \in [0, ..., K-1], \tag{4.27b}$$
$$u(k+1) = u(k) + \Delta u(k), \tag{4.27c}$$
$$\xi(k) \in \mathbb{X}, k \in [0, ..., K-1], \tag{4.27d}$$
$$u(k) \in \mathbb{U}, k \in [0, ..., K-1], \tag{4.27e}$$
$$\xi_0 = \xi \tag{4.27f}$$

$\boldsymbol{\xi} = [\xi(0), ..., \xi(K)]$ is the state sequences obtained by applying the control sequence $\boldsymbol{u} = [u(0), .., u(K-1)]$ to the system starting from the initial state ξ_0.

The MPC controller not only performs path tracking but also provides obstacle avoidance for obstacles on the reference path, which are detected using lidar sensors. The cost function $J_{\Delta U}$ considers both the centerline tracking and obstacle avoidance. Here, $\eta = [\dot{x}, \dot{\psi}, \dot{e_\psi}, \dot{e_y}]$, and $\eta_{ref_t} = [\dot{x}ref, \dot{\psi}, 0, 0, \dot{\psi}', \dot{x}ref]$. η_k and η_{ref_t} indicate the tracking states' deviation from the reference in the center of the lane, while Q, R, and S are weighting matrices.

To achieve this, we formulate the MPC controller as a finite-dimensional optimization problem in this study. As a result, we discretize the vehicle dynamics ξ using a sampling time T_s.

$$J(\xi, u_t, \Delta u_t) = \sum_{k=t}^{t+H-1} (||\eta_k - \eta_{ref_t}||^2 Q - ||u_k||^2 R - ||\Delta u_k||^2 S) \tag{4.28}$$

The controller must ensure that the vehicle's acceleration constraints are within the allowable margins to maintain the feasibility of the predictive control strategy. Furthermore, constraints are imposed to ensure that the vehicle stays on the road and moves in a forward direction.

$$\begin{cases} a_{min} < a < a_{max} \\ a_{min} < a < a_{max} \\ \frac{-\pi}{2} < e_\psi < \frac{\pi}{2} \\ -\rho < e_y < \rho \end{cases} \tag{4.29}$$

6 System architecture

The proposed system's general structure is illustrated in Figure 4.15. Inputs from the stereoscopic camera are processed by a Yolov5 model to detect objects. Moving and stationary objects are then tracked using a DeepSORT model. The model predictive control model uses the detected vehicle's position and velocity information to avoid obstacles.

7 Results and Discussion

A set of simulation scenarios was created to evaluate the proposed system architecture's efficiency in enabling a controlled vehicle to execute an overtake maneuver in different traffic conditions. The simulations were carried out using the Webots robotic simulator, which offers high-quality simulations with realistic visuals. Sumo, a tool that simulates urban mobility and analyzes road traffic and traffic management systems, was used to generate traffic flow. The objective of these simulations was to assess each architecture component's performance.

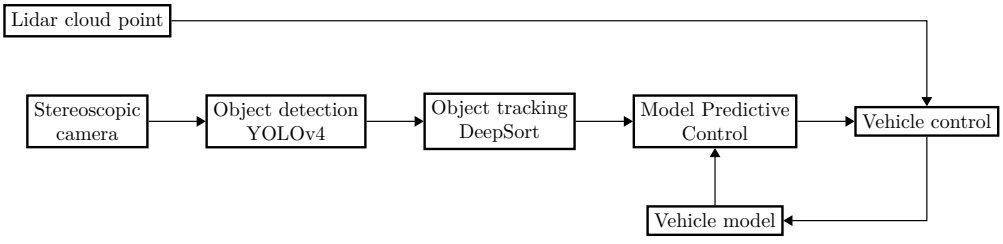


Figure 4.15: System architecture.

Table 4.2: Hyperparameters values.

Parameter	Value
m	1950 Kg
I_z	3700 kg.m^2
l_f	1.100 m
l_r	1.500 m

7.1 Hardware requirements

To simulate the vehicle's abilities, a stereoscopic camera is installed at the vehicle's front to detect lanes and provide initial trajectory information, such as the road's curvature, number of lanes, and size. Multiple Ibeo Lux 4-layer Light Detection and Ranging (LiDAR) devices are utilized to detect surrounding vehicles. With a range of 200 meters and a field of view of 110 degrees, these devices can provide 680 points per layer per scan, providing a complete view of the surrounding environment. The lidar devices accurately locate both stationary and moving objects on the road.

7.2 Hyperparameters

The simulation is based on the assumption that the vehicle is already operating at its maximum speed of 90 km/h on the rightmost lane. The MPC controller's horizon is set to N=20, meaning that the controller predicts 20 states into the future. While increasing the number of predicted states can enhance performance when dealing with faster vehicles, it also increases computation time. In this simulation, using N=20 yielded satisfactory results and allowed for real-time computation. The parameter values used in the simulation are outlined in Table Table 4.2.

7.3 Object detection and tracking performance

Performance evaluation of detection and tracking outcomes is carried out using established methods such as Multiple objects tracking precision (MOTP), Multiple object tracking accuracy (MOTA), Multiple Object Detection Accuracy (MODA), and Multiple Object Detection Precision (MODP) [156]. MOTP represents the average difference between true positives and their corresponding ground truth targets, indicating the tracker's object position estimate accuracy. MOTA measures the tracking algorithm's accuracy by considering false positives, false negatives, and identity switches. MODA and MODP evaluate object detection precision and accuracy by comparing predicted bounding boxes to ground truth data. Section Table 4.3 further analyzes the performance of object tracking in simulated scenarios. Figure 4.16 displays an instance of an image capture from the stereoscopic camera's right camera, while Figure 4.17 shows the outcome of object detection tracking and speed estimation. The oncoming vehicle's type and speed can affect object detection performance.

8 Test scenarios

We evaluated the performance of the complete system through multiple tests using various scenarios and road layouts to gain a detailed understanding of the strengths and weaknesses

Table 4.3: Object tracking performance.

Benchmark	**M**OTA	**M**OTP	**M**ODA	**M**ODP
Car	82.55%	78.32%	79.23%	76.85%
Truck	72.45%	73.35%	74.62%	71.85%

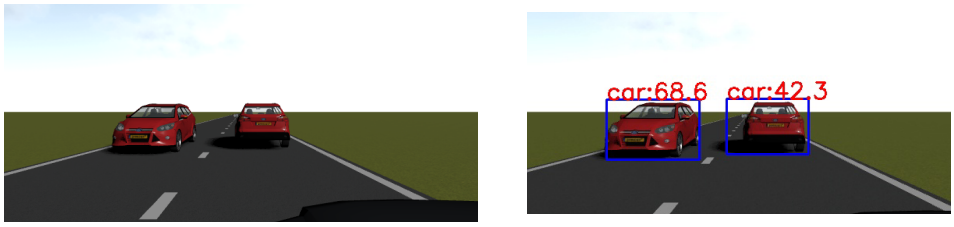


Figure 4.17: Object detection and velocity estimation.

of the proposed system. We repeated each scenario N=100 times with randomized traffic flow seed and recorded the success of the overtake based on factors such as leading and oncoming vehicle speed, fog, and safety margin violations. We also tested the controller's performance when the leading vehicle made random acceleration or deceleration during one of the three stages of the maneuver to determine how the controller handles variability in other vehicles' driving behavior.

1. Leading vehicle overtake The first scenario, depicted in Figure 4.18, involved the proposed system conducting an overtake of a leading vehicle traveling at a constant speed between 60 to 80 Km/h without any interference from other vehicles or obstacles. As there were only two lanes in the simulation, the MPC controller computed an alternative path while following the constraints outlined in Equation 4.29 when the vehicle detected a leading vehicle. Since the maneuver was performed on a two-lane road, the vehicle conducted a left-side overtake while keeping a safe distance from the overtaken vehicle to account for any variations or errors in its driving. After completing the overtake, the ego vehicle returned to the right lane.

The speed profile of the ego vehicle during the overtake is depicted in Figure 4.19. The ego vehicle maintained its maximum velocity of 80 km/h since there were no oncoming vehicles. However, during the lane-changing phase of the overtake, the MPC controller adjusted the ego vehicle's speed to maintain its position on the curved path.

Figure 4.20 displays the success rate of the maneuver and the percentage of safety margin violations across various speeds of the leading vehicle. It can be seen that in the absence of any conflicts with other vehicles, the system carries out a safe overtake with a success rate of up to 99.5% at low speeds and 92% at high speeds. Additionally, we can deduce that the presence of fog reduces the accuracy of the system, this could be due to the delayed detection of vehicles by the camera and lidar sensors caused by the presence of fog.

2. Overtaking in the presence of an oncoming vehicle from the front In the second scenario, the system performs an overtake of a leading vehicle while an oncoming

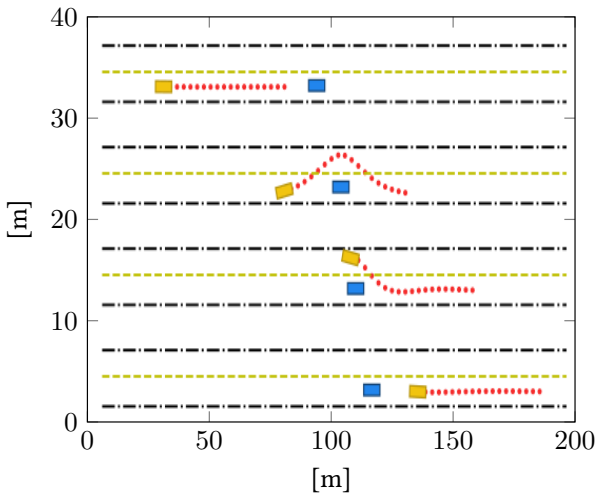


Figure 4.18: Simulation of a right-side passing maneuver showing a bird's-eye view.

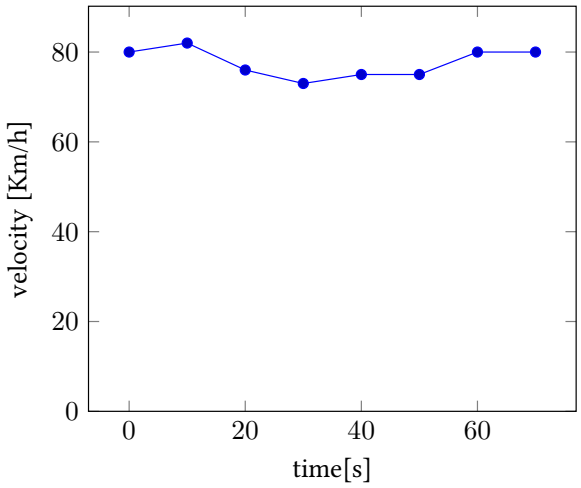


Figure 4.19: Vehicle'speed profile during the simulation.

vehicle is present, as shown in Figure 4.21. The MPC controller evaluates the feasibility of the overtake and whether it can be completed without collision with the oncoming vehicle. In this scenario, two cases were observed. Depending on the speed of the oncoming vehicle and leading vehicle, the MPC will either proceed with the overtake by accelerating to the appropriate speed within the predefined acceleration and speed margins, or it will decelerate and delay the overtake until the oncoming vehicle passes.

Figure 4.22 illustrates the speed profile of the vehicle during a simulation of an overtake in which the overtake is possible. The changes in velocity can be attributed to the MPC controller's search for the optimal time window to perform the overtake. In Figure 4.23 and Figure 4.24, it can be seen that the system has a higher chance of successfully completing the overtake if the overtake is delayed until the conflict with the oncoming vehicle is resolved. Additionally, the presence of fog has a greater

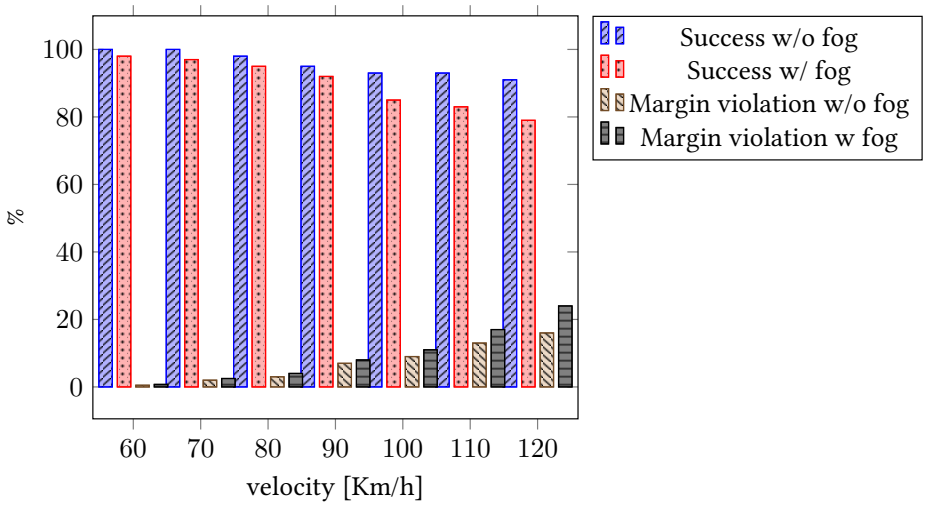


Figure 4.20: Successful passing and safety margin violations based on the speed of the lead vehicle, both with and without fog present.

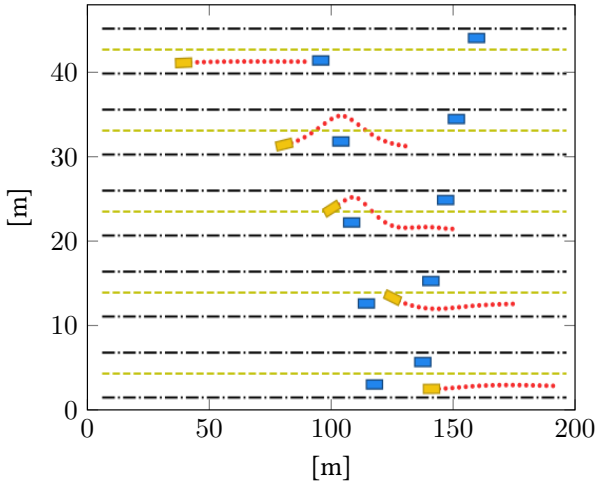


Figure 4.21: Bird's-eye view simulation of a right-side passing maneuver involving one vehicle moving in the same direction as the vehicle being simulated, and another incoming vehicle.

impact on the success of the maneuver in this scenario.

3. Overtaking on a curved road: The third scenario, shown in Figure 4.25, involves the system performing an overtake of a leading vehicle while an oncoming vehicle is present on the opposite lane of a curved road. The curvature of the road presents a challenge as it limits the angle of vision, making it difficult to detect vehicles using lidar sensors. This scenario is considered the most problematic type of maneuver.

In Figure 4.26, it shows the speed profile of the ego vehicle during two cases (immediate and postponed overtake), similar to previous scenarios we can see the MPC model attempt to plan for the optimal overtaking window depending on the position of other vehicles by adjusting the speed of the ego vehicle. Figure 4.27 describes the

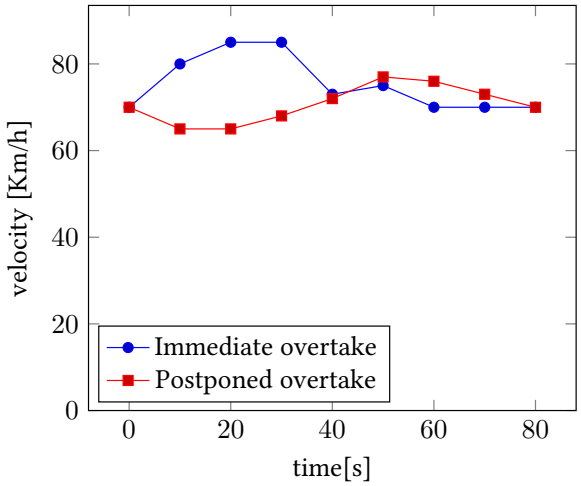


Figure 4.22: Speed of the vehicle being simulated during two scenarios: 1) immediate passing, 2) delayed passing.

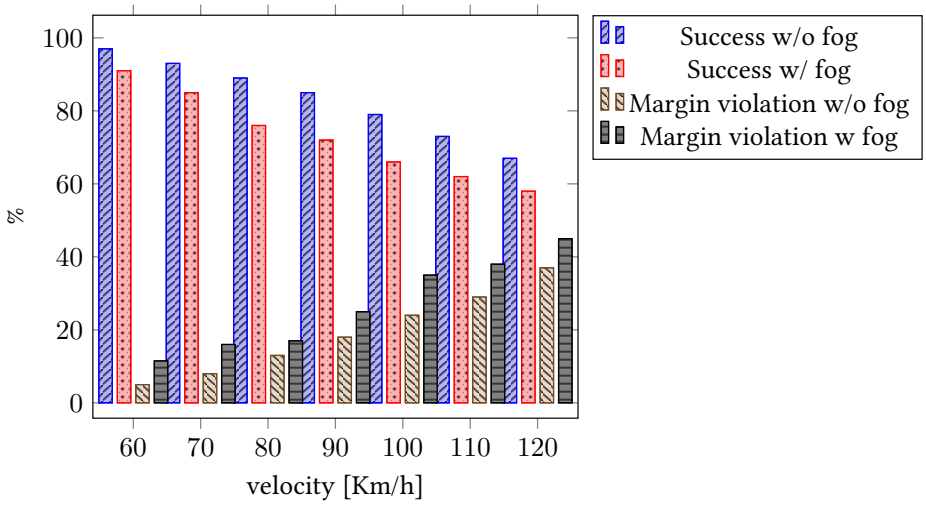


Figure 4.23: Successful passing and safety margin violations based on the speed of the lead vehicle during an immediate passing maneuver.

performance of the proposed system depending on the overtaking vehicle's speed and distance, where we can notice that the system has better performance when the vehicle is detected early. Therefore, better lidar and camera sensors are essential to the performance of the system.

4. Overtaking of a leading vehicle changing speed: In the fourth scenario, the system will attempt to perform overtake a leading vehicle. However, unlike in previous scenarios, the leading vehicle will change its speed during the overtake, either by accelerating or decelerating. The MPC controller will recalculate the appropriate path each time step, and if the leading vehicle's change of speed makes the overtake impossible, the MPC controller will either re-accelerate or terminate the overtake and return to the original lane. Figure 4.29 illustrates the percentage of successful

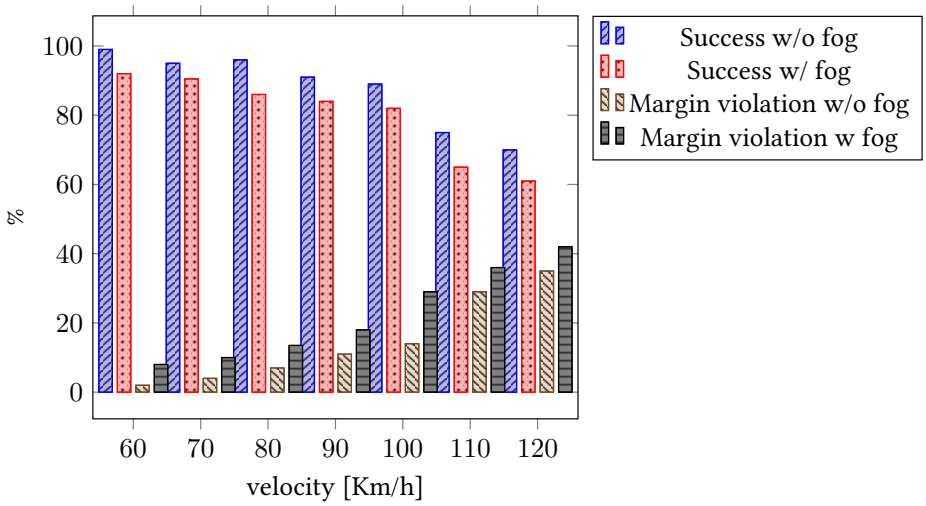


Figure 4.24: Successful passing and safety margin violations based on the speed of the lead vehicle during a postponed passing maneuver.

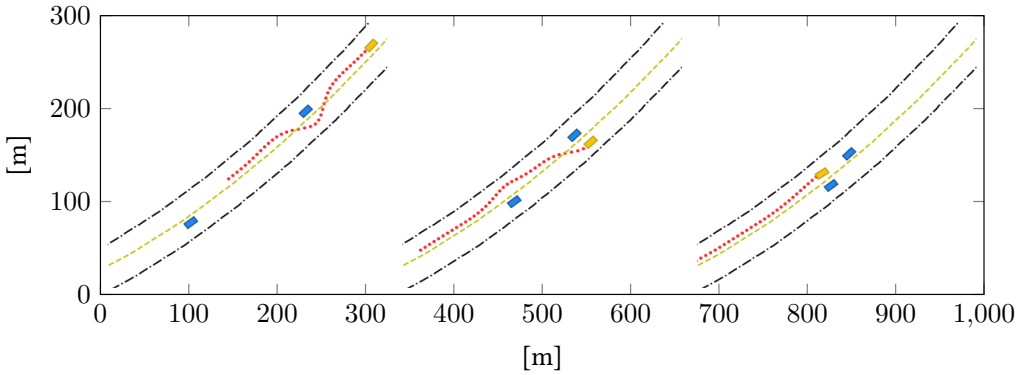


Figure 4.25: Simulation of a right-side passing maneuver viewed from above on a curved road.

maneuvers under different behaviors from the leading vehicles (acceleration or deceleration). It can be seen that the vehicle has a high chance of performing a successful maneuver, especially if the leading vehicle accelerates in the first stage or decelerates in the 3rd stage since at that point the maneuver is almost complete.

5. Managing mid-overtake speed changes. In the fifth scenario, the system will attempt to perform an overtake of a leading vehicle while taking into account the potential change in speed of the leading vehicle during the maneuver. The MPC controller will recalculate the appropriate path in real-time and make decisions on whether to continue or cancel the overtake based on the leading vehicle's behavior. Figure 4.30 shows that if the leading vehicle accelerates in the first stage of the overtake, the system has a higher chance of successfully completing the maneuver without collision. On the other hand, if the leading vehicle decelerates in the first or third stage, the MPC controller will often be able to cancel the overtake. However, if the leading vehicle changes its speed later in the overtake, the MPC controller may not be able to finish the maneuver successfully.

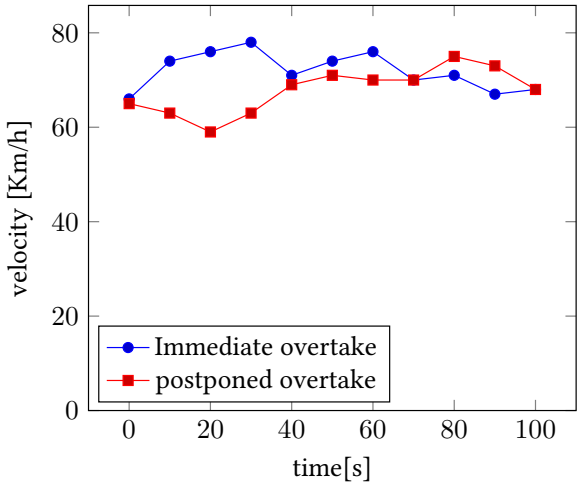


Figure 4.26: Speed of the vehicle being simulated during two scenarios: 1) immediate passing, 2) delayed passing.

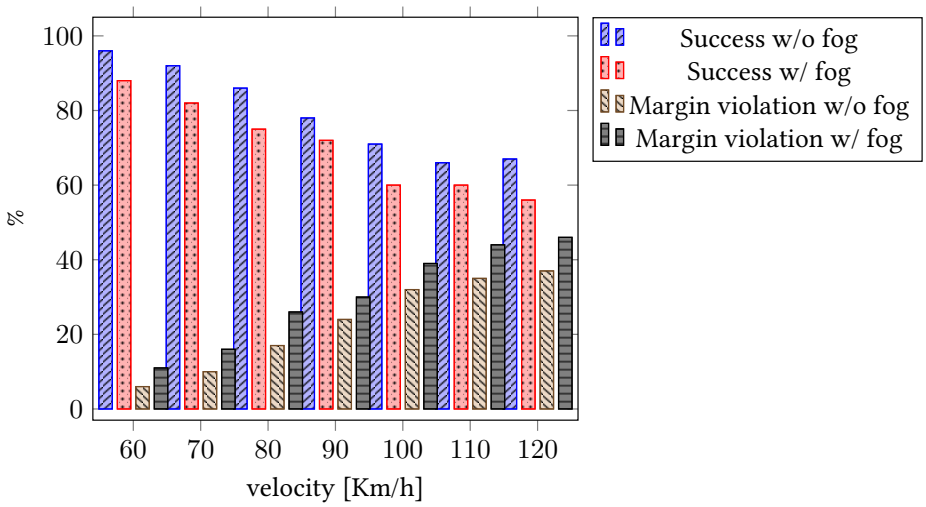


Figure 4.27: Successful passing and safety margin violations based on the speed of the lead vehicle during an immediate passing maneuver.

9 Conclusion

The main objective of the proposed system is to enable safe and efficient overtaking of leading vehicles in diverse road and traffic conditions. It achieves this by using a combination of lidar sensors and stereoscopic cameras to detect road architecture and other vehicles on the road. The system is built on the Model Predictive Control (MPC) approach, which predicts future system states based on a model of its components. To represent fast maneuvers like overtaking, a dynamic bicycle model is used. The system underwent testing in different scenarios, including straight and curved roads under varying weather conditions, with a specific focus on fog's effect on the system's performance. The findings indicate that the system can conduct a successful overtake while adhering to predefined

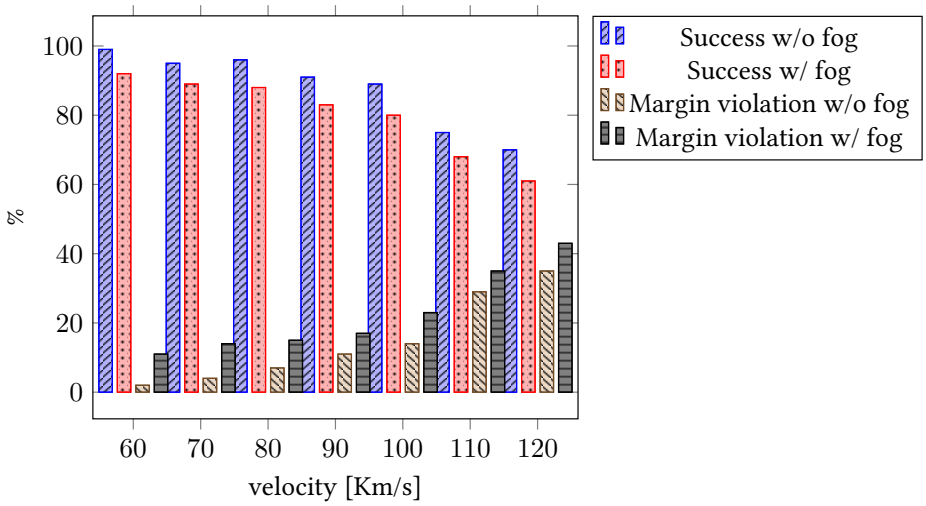


Figure 4.28: Successful passing and safety margin violations based on the speed of the lead vehicle during a delayed passing maneuver.

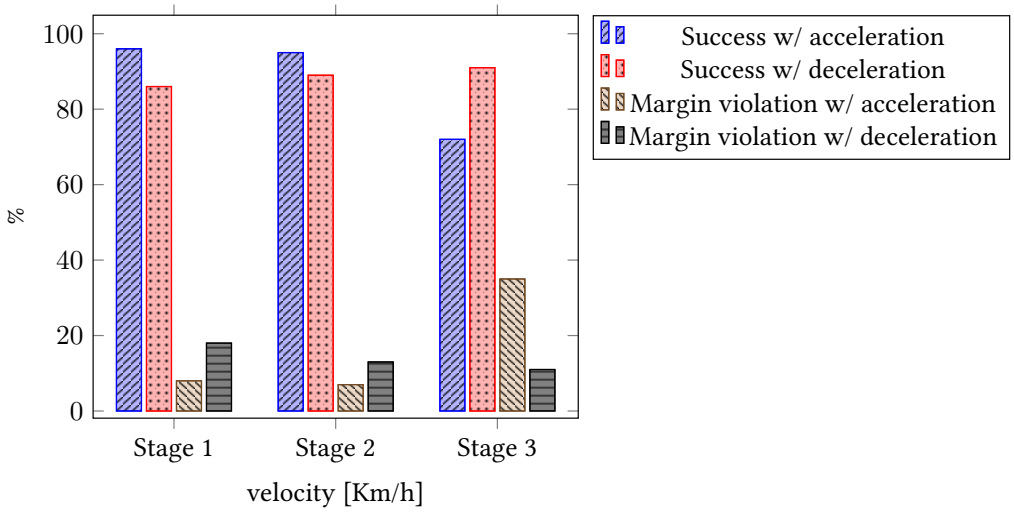


Figure 4.29: Successful overtake and safety margin violation when the leading vehicle performs a speed change mid-overtake.

constraints on speed, acceleration, and path, particularly at lower speeds. However, the performance of the system reduced as vehicle speeds increased, and the leading vehicle's behavior affected the maneuver's outcome. Unexpected speed changes by the leading vehicle reduced the MPC-controlled vehicle's efficiency, particularly in the second stage of the overtake.

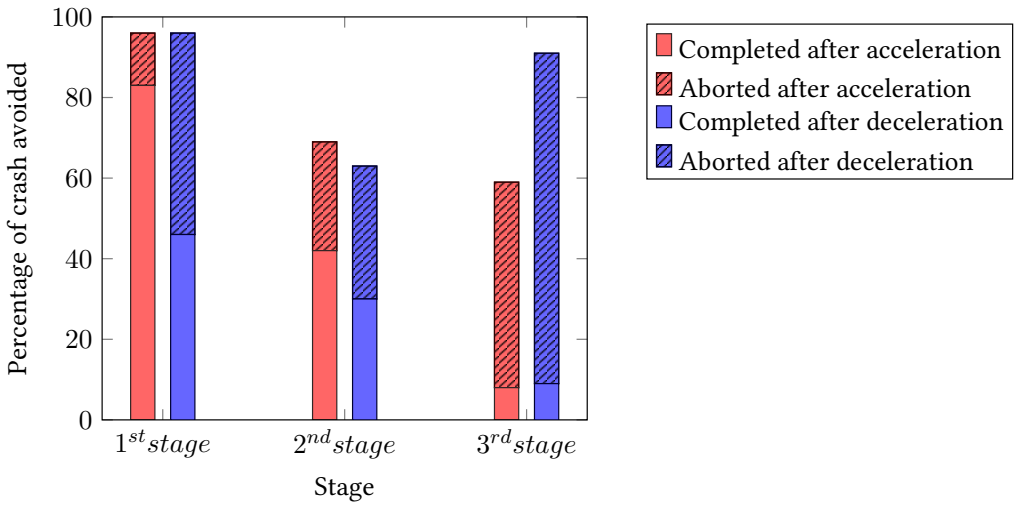


Figure 4.30: Successful maneuvers with leading vehicle's random speed change during overtaking

Conclusion and Future Directions

Academics worldwide have been focused on traffic control since the start of the century, driven by the desire to improve the driving experience, decrease traffic congestion, and prevent accidents. However, achieving this goal has been difficult due to the large and intricate nature of city traffic. This thesis aims to tackle the issue of traffic control from various perspectives and suggest a collection of systems that can address prevalent and difficult problems that impact both human drivers and autonomous vehicles.

To address the problem of traffic congestion, which has become a daily occurrence for drivers, we began this thesis by exploring optimal traffic routing as a solution. We proposed a routing system using deep learning architecture that was able to improve the average trip time in a dynamic traffic grid. However, as the grid and number of vehicles grew, the system became less effective. To overcome this challenge, we enhanced our approach by dividing the grid into smaller sub-networks and using Q-routing as the routing algorithm. The improved system demonstrated superior results and remained effective even in large city-sized networks.

Our research looked into the possibility of eliminating traffic signals at intersections by implementing a multi-agent system to coordinate the movement of vehicles. This would allow autonomous vehicles to navigate busy intersections by communicating with other vehicles. However, the system did not take into account pedestrians, which will be the focus of future work.

Lastly, the thesis examined the hazardous task of vehicle overtaking, which poses a significant risk to both novice and experienced drivers and causes numerous deaths annually. A system was developed for autonomous vehicles to safely overtake other vehicles using model predictive control, which utilizes a combination of camera and lidar sensors for stereoscopic vision. This system employs Deepsort and Yolo for gathering information about the environment, such as lane width, lane center and distance from neighboring vehicles. The results showed that the system can perform safe overtaking under various driving scenarios and weather conditions.

However, the high cost of the lidar sensors used in the system is a significant concern regarding its practicality and feasibility. In order to address this issue, future work should explore the possibility of using less expensive radar sensors as an alternative. While lidar

sensors are highly precise, they are also relatively costly. Radar sensors, on the other hand, can provide similar functionality at a lower cost. Therefore, it would be worth investigating the feasibility of using radar sensors in place of lidar sensors in order to reduce the overall cost of the system and make it more accessible to a wider range of users. Additionally, future work should also investigate other methods to reduce the cost of the system, and make it more financially accessible for the end-users

APPENDIX A

Résumé étendu en français

1 Introduction

Le contrôle de la circulation est, depuis le siècle dernier, à l'épicentre de l'intérêt des chercheurs, encouragé par le besoin émergent d'améliorer l'expérience de conduite, la réduction des embouteillages et l'élimination des accidents de la circulation. Et en raison de l'augmentation de la puissance de calcul et des progrès technologiques, il est devenu possible de développer des solutions de contrôle plus robustes capables de gérer des tâches de trafic complexes qui étaient autrement loin de la portée des systèmes de conduite autonome.

Les embouteillages sont connus pour allonger les trajets et augmenter la pollution; ainsi, en offrant une solution de routage optimale, nous visons à améliorer l'expérience de conduite. Par conséquent, dans cette thèse, nous avons commencé par considérer l'acheminement optimal du trafic comme une solution au problème de congestion du trafic puisqu'il est devenu un ennui quotidien pour les conducteurs dans la plupart des villes modernes.

De plus, afin d'atteindre une autonomie complète, une stratégie de routage fiable et rapide est nécessaire pour calculer les chemins optimaux. Pourtant, les algorithmes statiques et centralisés existants ont d'énormes limitations les empêchant de résoudre le problème de congestion, en particulier compte tenu de la taille des villes modernes et du volume de trafic quotidien.

Par conséquent, l'objectif de cette thèse est d'aborder le problème du contrôle du trafic et de proposer un ensemble de systèmes capables de résoudre des problèmes courants mais difficiles qui affectent à la fois les conducteurs et les systèmes de conduite autonome.

2 Le problème du routage optimal du trafic

Les embouteillages ont augmenté de manière alarmante dans les villes modernes, malgré les efforts soutenus des gouvernements et des urbanistes pour financer et construire de plus grandes infrastructures routières. Cependant, à mesure que la demande de trafic continue d'augmenter, il devient de plus en plus évident que la capacité du réseau de transport sera toujours insuffisante pour gérer le flux croissant de véhicules. Un autre aspect crucial à

prendre en compte dans la planification des transports est le concept de demande induite[7], où toutes les routes ou voies supplémentaires ont tendance à se remplir rapidement à pleine capacité. En d'autres termes, à mesure que les routes grandissent, elles attirent toujours plus de trafic pour remplir le nouvel espace. En fait, la plupart des villes ont déjà atteint le point où le flux de trafic est supérieur à la capacité de toutes les voies supplémentaires.

En outre, l'expansion récente de la taille des villes modernes et la disponibilité de véhicules pour un nombre croissant de personnes de la classe moyenne ont entraîné une pression supplémentaire sur le réseau de circulation, obligeant des millions de personnes à passer des heures chaque jour coincées dans des embouteillages. Ces trajets plus longs ont été associés à divers problèmes de santé allant de symptômes physiques tels que des maux de tête, des douleurs au dos, des problèmes digestifs et de l'hypertension artérielle, à des troubles mentaux tels que des troubles du sommeil, de la fatigue et des difficultés de concentration[8].

Par conséquent, Il est crucial d'optimiser l'utilisation des ressources et de réduire les longueurs moyennes des trajets, les temps de trajet et la consommation de carburant des véhicules pour résoudre les problèmes d'embouteillages. Pour y parvenir, un routage efficace du trafic est essentiel. Cependant, la tâche devient de plus en plus complexe pour les algorithmes statiques traditionnels lorsque la taille du réseau augmente, car les réseaux de trafic sont par nature très dynamiques et en constante évolution. Il est donc nécessaire de poursuivre les recherches pour concevoir des systèmes de planification et de routage de chemin plus performants et adaptatifs.

Les réseaux de circulation ont longtemps été limités par l'absence d'algorithmes de routage dynamique et de planification de chemin, se basant uniquement sur des caractéristiques statiques des routes, telles que le nombre de voies, la distance et les limites de vitesse, pour éviter et résoudre les embouteillages en acheminant le trafic vers des routes moins congestionnées. Cependant, avec l'augmentation du nombre de véhicules dans les zones urbaines, ces algorithmes ont peut-être atteint leur limite en raison de l'augmentation massive de l'espace état/action. Il est donc nécessaire de développer des algorithmes plus avancés et adaptatifs pour relever les défis de la circulation moderne.

2.1 Le système proposé

Cette thèse introduit un système de routage qui bénéficie de la puissance des algorithmes de Q-learning et du clustering spectral pour offrir une solution de routage décentralisée, dynamique et proactive. Les principaux avantages du système sont sa faible exigence de calcul puisque le processus de routage est réparti sur plusieurs petits nœuds. De plus, il peut s'adapter rapidement aux changements de la topologie du réseau en offrant une propagation rapide des informations de routage à travers les nœuds adjacents. Enfin, il fournira un routage en temps réel puisque le calcul du chemin est réalisé à l'aide de poids de chemin précalculés reflétant l'état de chaque chemin dans le réseau de trafic.

L'un des principaux obstacles auxquels sont confrontés le Q-Routing(subsection 2.1), en particulier, et le Q-Learning, en général, est la façon dont il souffre de la malédiction de la dimensionnalité[148], qui survient lorsque la taille de l'espace des solutions augmente de façon exponentielle à mesure que l'espace état/action augmente. Cette faille est causée par son utilisation d'une méthode de stockage tabulaire pour stocker ses estimations de routage, augmentant rapidement la taille du Q-Table à mesure que la taille du réseau augmente,

rendant l'utilisation du Q-Routing dans des réseaux urbains de taille réelle très peu pratique.

D'autre part, le Q-Routing souffre de la propagation lente des informations de trafic entre deux nœuds en raison de son mécanisme de mise à jour. Parce que dans le Q-Routing, le nouvel état d'un nœud n'est transféré au nœud adjacent que lorsque ce nœud envoie un véhicule à ce nœud. Sinon, il n'y aura pas de communication entre le nœud et ce voisin. Cette conception entraîne un délai de mise à jour de l'état du réseau proportionnel à la longueur du chemin (nombre de sauts), comme indiqué dans Figure A.1. En outre, cela peut même entraîner le transfert du trafic via un chemin déjà rompu dont l'état n'a pas encore atteint le nœud de transfert.

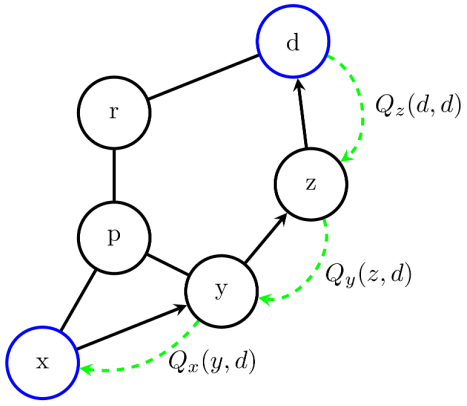


Figure A.1: Exemple de routage Q (les flèches pleines représentent les véhicules en cours de routage et les flèches en pointillés représentent la propagation de l'état du trafic).

Notre architecture proposée aura deux avantages; le premier est d'éviter les calculs inutiles lorsque le routage se produit dans le cluster local, réduisant ainsi la taille des Q-Tables, minimisant ainsi les exigences matérielles du système. Et deuxièmement, cela simplifiera le routage vers des nœuds distants hors des clusters en pré-affectant des nœuds d'entrée et de sortie à des clusters distants, ce qui minimisera le temps de calcul de l'algorithme.

L'architecture proposée comprend trois différents types d'agents intelligents et indépendants, chacun avec un ensemble de tâches spécifiques (Figure A.2).

2.1.1 Définition des agents

Agent de véhicule: Installé dans chaque véhicule et responsable de la communication (requête/réponse) entre le véhicule et les "agents de routage". Son objectif principal est de demander un itinéraire aux "agents de routage" chaque fois qu'il s'approche d'une intersection en fournissant sa destination souhaitée.

Agents de routage de cluster interne: Installé dans chaque nœud et ayant deux responsabilités principales:

1. Fournir un routage local aux destinations qui résident à l'intérieur du cluster interne à l'aide de son estimation de l'état de routage local fournie par:

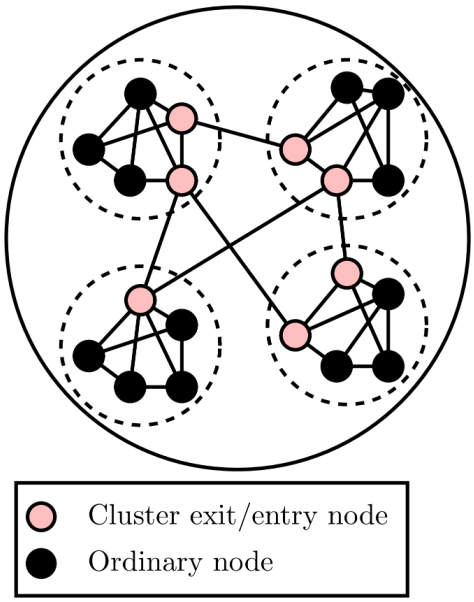


Figure A.2: Regroupement du réseau de trafic.

$$y = \arg \min_{z \in N(y)} Q_x(d, z) \tag{A.1}$$

2. Si la destination se trouve en dehors du cluster local, elle fournit la route vers le meilleur nœud de sortie du cluster local.

Agent de routage de cluster externe : Installé dans chaque nœud de sortie/d'entrée de cluster et responsable du routage vers des destinations en dehors du cluster local en acheminant vers le meilleur nœud d'entrée pour le cluster de destination.

2.1.2 Algorithme de routage

Le processus d'algorithme de routage (décrit dans Figure A.3 et Algorithm 2) fonctionne en permettant à plusieurs types d'agents de routage d'être installés dans le même nœud en fonction de la position du nœud dans la topologie du réseau.

2.2 Expériences et résultats

L'efficacité de l'architecture proposée a été implémentée et testée sur le réseau de la ville "Fès, Maroc", une ville d'une superficie de $96.5km^2$. Fès est l'une des villes les plus peuplées du Maroc. Le réseau utilisé comprend 11902 nœuds et 18167 sommets et a une longueur moyenne du plus court chemin de $56,927$. Le réseau de la ville Figure A.4a est extrait de "OpenStreetMap" [149] et d'abord regroupé en clusters de 10 (peut être modifié en fonction de la taille et des caractéristiques de la ville) en utilisant le clustering spectral comme décrit ci-dessus.

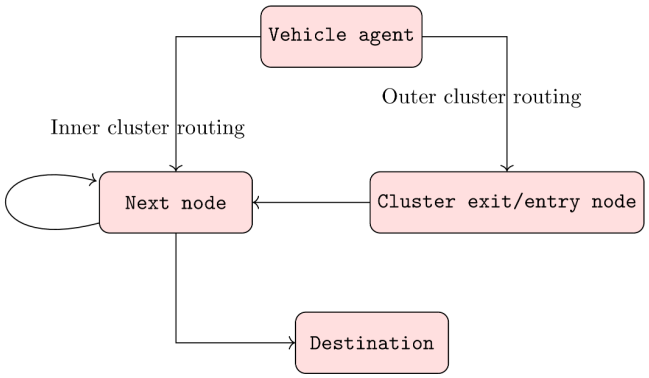


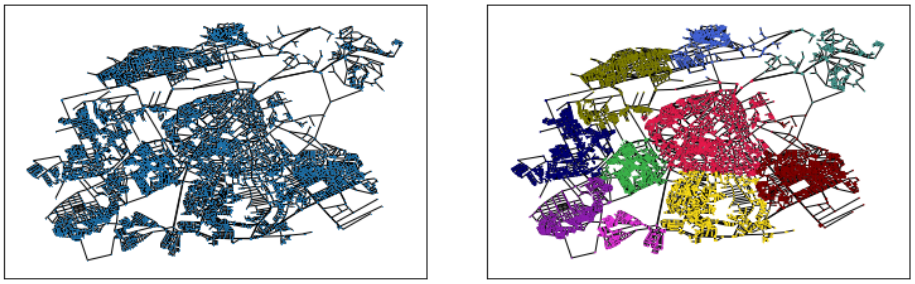
Figure A.3: Le schéma du processus de routage.

Algorithm 2 Algorithme de routage.

Ensure: Route vehicle v from a node s to a node d

while not at destination d **do**
  if v at exit/entry node **then**
    route v to destination's cluster
  end if
  if Outer routing **then**
    route v to cluster exit node
  end if
  if Inner routing **then**
    route v to destination d
  end if
end while

Le résultat du clustering est montré dans Figure A.4b, où l'on remarque que le réseau était bien clusterisé et que les clusters sont séparés par un petit nombre de nœuds qui peuvent servir de nœuds d'entrée/sortie entre les différents clusters homogènes.



(a) **(b)**

Figure A.4: La ville de Fès, Maroc. (a) avant regroupement et (b) après regroupement en 10 groupes distincts.

Après avoir réalisé un clustering adéquat du réseau, nous avons effectué plusieurs

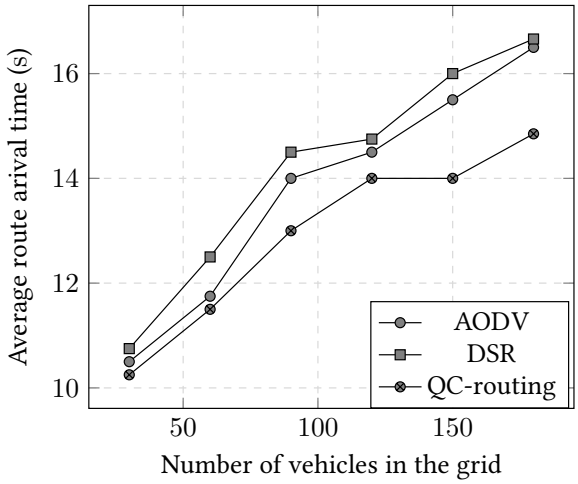


Figure A.5: Temps moyenne d'arrivée des véhicules dans des conditions normales de circulation.

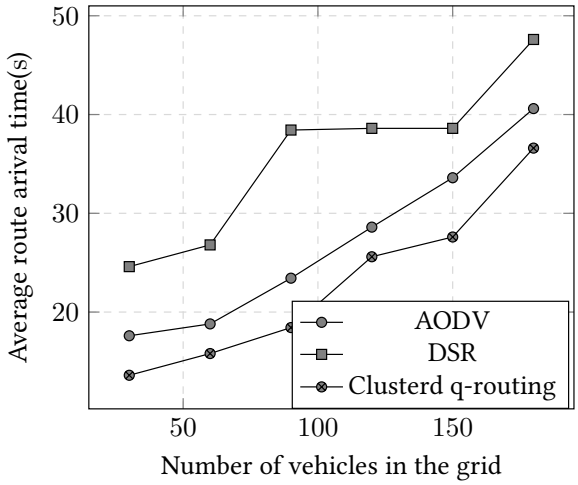


Figure A.6: Temps d'arrivée moyen des véhicules dans des embouteillages générés de manière stochastique.

scénarios de routage simulés; la première est de comparer l'architecture proposée avec les algorithmes de routage (AODV) et (DSR) en conditions normales de trafic Figure A.5. La seconde consiste à les comparer dans une congestion ajoutée dynamiquement en plus des changements d'état aléatoires du réseau pour tester comment les trois algorithmes géreront les changements dynamiques du réseau Figure A.6. La congestion introduite consiste à bloquer aléatoirement certaines routes pendant un certain temps, forçant des réacheminements à être effectués.

Dans les deux scénarios, nous avons étudié le temps de trajet global des véhicules à l'intérieur du réseau et pouvons clairement voir que les algorithmes proposés surpassent (AOSV) et (DSR) et ont donné un meilleur temps de trajet pour l'ensemble du réseau; les meilleurs résultats peuvent s'expliquer par la façon dont l'algorithme proposé peut prédire l'accumulation de congestion avant l'arrêt complet d'un itinéraire et ajuster les

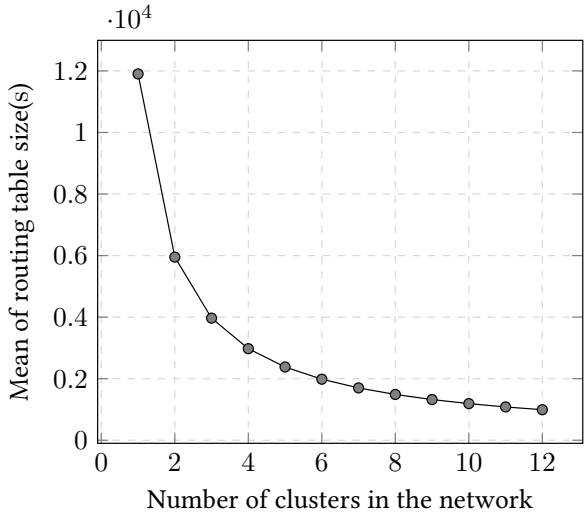


Figure A.7: Taille moyenne de la table de routage.

itinéraires en conséquence. Compte tenu de leur nature stochastique, ces résultats rendent les algorithmes proposés utilisables dans des scénarios réels. De plus, nous pouvons remarquer dans Figure A.7 qu'en augmentant le nombre de clusters, nous constatons une diminution significative de la taille de la table, ce qui signifie une utilisation efficace de la mémoire système.

2.3 Conclusion

L'augmentation des flux de trafic et de la congestion dans la plupart des villes a créé une demande pour des solutions de routage efficaces qui peuvent bien fonctionner même lorsque la taille du réseau augmente. Cependant, la plupart des modèles existants souffrent lorsqu'ils sont utilisés dans des scénarios de trafic réels en raison de leur incapacité à gérer les changements dynamiques et fréquents de l'état du réseau ou les blocages soudains de routes dans la topologie du réseau.

Dans cette thèse, nous avons présenté une architecture pour fournir une solution de routage optimale dans une zone urbaine très dense en partitionnant le réseau en utilisant le clustering spectral en plusieurs clusters avec un nombre minimum de connexions. Et deuxièmement, en séparant la politique de routage en deux phases, le routage de cluster interne et le routage de clustering externe utilisant le Q-routing, cette approche peut réduire considérablement les ressources matérielles et le temps de calcul puisque nous réduisons l'espace d'action.

Le partitionnement du réseau en plusieurs clusters crée plusieurs zones homogènes avec un trafic concentré, minimisant la nécessité de comparer les routes avec des nœuds distants (intersection).

Enfin, nous avons effectué une simulation basée sur deux scénarios. Le premier scénario correspond à des conditions de circulation normales. La seconde concerne la congestion stochastique pour tester la capacité du système proposé à s'adapter à la nature aléatoire de la circulation des véhicules. Au final, nous avons conclu que l'algorithme proposé

surpassait les autres algorithmes (AODV et DSR) et s'adaptait aux changements de niveau de congestion du trafic.

3 Le problème du dépassement des véhicules

Le domaine des véhicules autonomes a connu de nombreuses avancées, tant théoriques qu'empiriques. Cependant, l'un des défis restants dans la tâche de conduite, auxquels sont confrontés les systèmes autonomes et les conducteurs humains, est la manœuvre de dépassement, qui n'a pas connu de percée significative dans la recherche universitaire, ce qui la rend loin d'être utilisée dans les véhicules grand public.

Ce manque de progrès peut être attribué à de nombreux facteurs, mais ils peuvent généralement être divisés en deux catégories : les problèmes liés à la perception (par exemple, la détection) et les problèmes algorithmiques (par exemple, la planification). Le problème de perception de l'environnement est lié à la façon dont un système peut percevoir avec précision son environnement. Ces connaissances sont théoriquement facilement obtenues à partir d'un ensemble de capteurs comme les LiDAR, les caméras, les radars, les GPS, etc. Cependant, la tâche de traitement et d'interprétation de ces informations est relativement difficile en fonction des objectifs du système. Cependant, les problèmes liés à la planification sont beaucoup plus difficiles à résoudre et ont nécessité des efforts de recherche importants pour développer des algorithmes de routage, de navigation et de contrôle optimaux, ainsi que des efforts liés au test et à l'optimisation de ces algorithmes. Gardant à l'esprit que la plupart de ces algorithmes nécessitent l'utilisation de l'état des véhicules équipés de matériel coûteux.

L'évaluation de la sécurité des dépassements doit tenir compte de plusieurs facteurs tels que a) La présence de véhicules venant en sens inverse à l'avant et à l'arrière. b) Vitesse relative et distance des véhicules environnants. c) Le nombre de véhicules de tête. d) L'écart entre le véhicule de tête et le véhicule hôte). La plupart de ces facteurs sont difficiles à estimer avec précision pour un conducteur humain. Par conséquent, il est tout à fait déraisonnable de faire confiance aux conducteurs pour évaluer la sécurité de l'exécution de ces tâches en se fiant uniquement à leurs capacités.

Dans ce travail, nous utilisons un système basé sur le modèle de contrôle prédictif pour le problème de dépassement. Grâce à ce système, un véhicule autonome effectuera un dépassement en toute sécurité d'un véhicule en présence d'un véhicule venant en sens inverse sur la deuxième voie. Le système utilise une caméra stéréoscopique frontale combinée à un lidar pour percevoir l'environnement et recueillir des informations sur la route et la position des obstacles fixes et mobiles, en plus de fournir des mesures de trajectoire préliminaires, y compris le nombre de voies, la largeur et la courbure de la route. Le flux d'images est utilisé comme entrée pour le système de détection et de suivi d'objets composé de deux modèles d'apprentissage automatique YoloV5 et DeepSORT pour calculer les distances et la vitesse relatives entre le véhicule hôte et les véhicules dans le flux de trafic venant en sens inverse ainsi que tout objet stationnaire sur la route . Le système garantira que la manœuvre de dépassement ne commencera que si le système peut entièrement exécuter un dépassement sûr et complet. Contrairement aux autres solutions existantes [157], le système utilise un modèle de vélo dynamique complet pour représenter avec précision la dynamique du véhicule, notamment lors des manœuvres à grande vitesse.

3.1 Modèle de véhicule

3.1.1 Modèle de véhicule dynamique dépendant du temps

L'une des étapes essentielles de la création d'une architecture de contrôle de véhicule performante et résiliente consiste à sélectionner un modèle de système approprié qui équilibre la précision du modèle et la simplicité de calcul. Dans le domaine des véhicules autonomes, plusieurs modèles de véhicules sont utilisés pour la conception et la simulation des commandes du système, chacun avec ses avantages et ses inconvénients en fonction du problème visé. Ces modèles vont du modèle de masse ponctuelle linéaire, du modèle de masse ponctuelle non linéaire, du modèle de vélo cinématique ou dynamique au modèle complet à quatre roues. Pour la plupart, ils ont été suffisamment étudiés dans la littérature [158] et [159]. De plus, il est nécessaire de sélectionner les variables de contrôle appropriées en fonction des spécificités de la manœuvre cible et des contraintes d'environnement pour établir un modèle capable de décrire avec précision le comportement du véhicule. Cependant, si le modèle est trop complexe, cela affectera les performances en temps réel de l'algorithme de contrôle.

Dans la plupart de la littérature, un modèle cinématique de véhicule très simplifié est souvent choisi pour modéliser les mouvements à basse vitesse et les angles de braquage de la manœuvre du véhicule car il offre généralement des résultats satisfaisants. Cependant, pour les manœuvres à grande vitesse telles que la manœuvre de dépassement, un modèle plus réaliste et exhaustif doit être choisi pour intégrer entièrement la dynamique complexe du mouvement du véhicule contrôlé et son interaction avec la route, comme les frottements des pneus qui peuvent généralement être négligé pour les contrôleurs de manœuvre à basse vitesse sans sacrifier la précision.

3.1.2 Modèle de vélo dynamique

Dans cette thèse, nous utilisons un modèle de vélo dynamique complet pour représenter la dynamique du véhicule, contrairement à d'autres tentatives dans la littérature. Le modèle de vélo dynamique offre une représentation réaliste des forces appliquées au véhicule, telles que les frottements des pneus et les angles de dérapage, qui offre une description robuste de l'état actuel du véhicule et permet par conséquent une inférence précise des états futurs du véhicule. De plus, le modèle de vélo a moins d'états qu'un modèle complet à quatre roues sans perte de précision du modèle. Le modèle de vélo regroupe les axes gauche et droit dans une roue unique située au centre de l'essieu avant (resp. essieu arrière). Table A.1 présente les notations utilisées dans le reste de la thèse.

Figure A.8 décrit les différentes forces agissant sur le cadre fixe de la carrosserie du véhicule, telles que la force longitudinale du pneu F_x et la force latérale du pneu F_y, les forces dans le cadre fixe du pneu (F_{l*} et F_{c*}), les vitesses de rotation et de translation, et

l'angle de braquage des roues avant δ.

Table A.1: Notations.

Symbole	Nom
F_{xf},F_{yf}	forces des pneus avant
F_{xr},F_{yr}	forces des pneus arrière
$\dot{x}$,$\dot{y}$	coordonnées de vitesse dans le cadre fixe du corps
ψ	angle de cap (lacet), dans le référentiel inertiel
I_z	moment d'inertie autour de l'axe z
δ	angle de braquage des roues avant
α	angle de glissement
ρ	taux de glissement
C_α	rigidité en virage pour les pneus avant et arrière
C_ρ	rigidité longitudinale
r	rayon du pneu
ω	vitesse angulaire du pneu
l_f, l_r	distance du centre de masse aux pneus avant et arrière

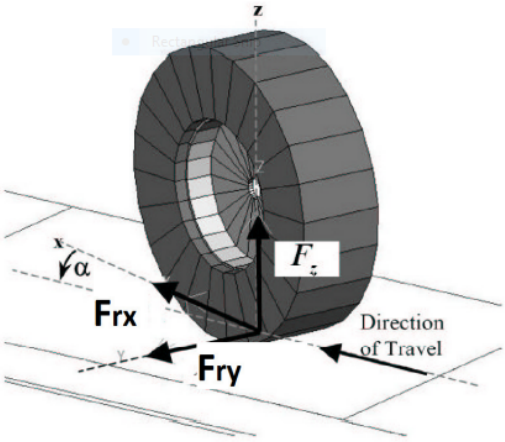


Figure A.8: Notation du modèle de vélo [150]

$$\ddot{x} = \frac{1}{m}(F_{xr} + F_{xf} + m\dot{\psi}\dot{y}) \tag{A.2a}$$

$$\ddot{y} = \frac{1}{m}(F_{yr} + F_{yf} - m\dot{\psi}\dot{x}) \tag{A.2b}$$

$$\ddot{\psi} = \frac{1}{I_z}(l_f F_{yf} - l_r F_{yr}) \tag{A.2c}$$

$\dot{\psi}$ est l'angle de lacet du véhicule, dans le repère inertiel, la vitesse angulaire du véhicule par rapport à l'axe global au centre de gravité (COG). Les forces longitudinales du pneu (F_{xf} et F_{xr}) sont générées lorsque le véhicule roule en ligne droite, lors de l'accélération et du freinage. Les forces latérales des pneus (F_{yf},F_{yr}) sont générées lorsque le véhicule change de direction dans les virages.

La position globale du véhicule peut être définie par:

$$\dot{X} = \dot{x}cos(\psi) - \dot{y}sin(\psi) \tag{A.3a}$$

$$\dot{Y} = \dot{x}sin(\psi) + \dot{y}cos(\psi) \tag{A.3b}$$

Le véhicule peut être contrôlé en suivant une trajectoire générée en suivant la ligne médiane de la voie la plus à gauche à l'aide de Equation A.2.

3.1.3 Forces des pneus

Pour obtenir un modèle complet de la dynamique du véhicule, il faut calculer les efforts des pneumatiques agissant sur le véhicule, d'autant plus que, dans notre implémentation, le véhicule effectue un mouvement à grande vitesse. Et comme les manœuvres de dépassement à grande vitesse sont entièrement limitées aux routes goudronnées, où l'angle de glissement α et le taux de glissement ρ ont de petites valeurs, un modèle de pneu linéaire peut être utilisé pour les forces latérales des pneus. Les forces de pneu peuvent être écrites à l'aide d'un modèle de pneu linéaire simplifié.

$$F_{yf} = C_\alpha \alpha_f \tag{A.4a}$$

$$F_{yr} = C_\alpha \alpha_r \tag{A.4b}$$

$$F_{xf} = C_\rho \rho_f \tag{A.4c}$$

$$F_{xr} = C_\rho \rho_r \tag{A.4d}$$

Le coefficient de raideur en virage C_α pour les pneus avant et arrière et la raideur longitudinale C_ρ peuvent être définis par :

$$\alpha_f = \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}} - \delta \tag{A.5a}$$

$$\alpha_r = \frac{\dot{y} - l_r\dot{\psi}}{\dot{x}} \tag{A.5b}$$

$$\rho_* = \begin{cases} \frac{r\omega - \omega_*}{r\omega} - 1 & \text{during braking} \\ 1 - \frac{r\omega}{r\omega - \omega_*} & \text{during acceleration} \end{cases} \tag{A.5c}$$

3.1.4 Modèle de véhicule à dynamique spatiale

Nous désignons le COG du véhicule par une paire de coordonnées spatiales relatives (e_ψ, e_y) où e_y est l'écart du véhicule par rapport à la position latérale sur une trajectoire courbe, e_ψ est l'écart du véhicule par rapport à l'axe de la trajectoire ρ, ϕ_s est le rayon du pneu et s la position du véhicule le long de l'axe de la route. Cette transformation de coordonnées permet de dériver la dynamique du véhicule le long de l'axe d'une voie. Le système de coordonnées curvilignes est indiqué dans Figure A.9.

$$\dot{s} = \frac{\rho}{\rho - e_y}(\dot{x}\cos(e_\phi) - \dot{y}\sin(e_\phi)) \tag{A.6}$$

La dérivée par rapport à s :

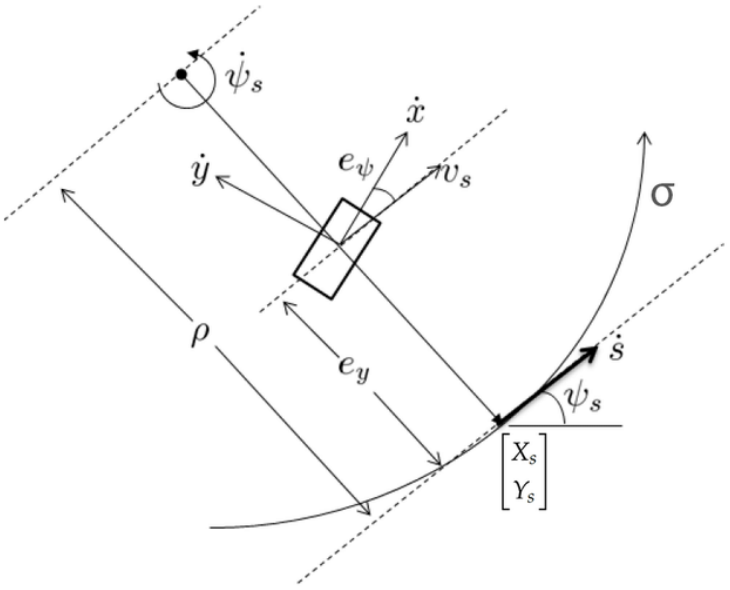


Figure A.9: Système de coordonnées curvilignes [150]

$$e'_y = (\dot{x}\sin e_\psi + \dot{y}\cos e_\psi)/\dot{s} \tag{A.7a}$$
$$e'_\psi = \dot{\psi}/\dot{s} - \psi'_s \tag{A.7b}$$

3.2 Contrôle prédictif du modèle

L'idée de base de MPC est de résoudre de manière répétée un problème de contrôle optimal pour l'état actuel du système sur un horizon fini H de N instants d'échantillonnage selon certains critères. Cependant, seule la première partie du signal d'entrée résultant est appliquée au système. A ce moment, le problème de contrôle optimal est résolu pour le nouvel état du système. Dans MPC, un modèle de véhicule est utilisé pour prédire son état futur. Sur la base de ces prédictions, à chaque incrément de temps t, un indice de performance est optimisé sous contraintes environnementales.

La première prédiction à partir de l'horizon temporel fuyant est appliquée au véhicule. Au pas de temps suivant $t+1$, une nouvelle prédiction est calculée à l'aide de son modèle sur un horizon de temps décalé H. Par conséquent, comme indiqué ci-dessus, il est impératif de choisir un modèle de système approprié qui décrit avec précision les états futurs du système en fonction de diverses entrées sur l'horizon H.

Comme indiqué ci-dessus, il existe de nombreuses approches basées sur les MPC dans la littérature. Cependant, ces systèmes ne sont généralement testés que dans des environnements de simulation 2D, où l'état de l'environnement est globalement connu du modèle, ce qui est irréaliste et ne fournit pas une référence objective de la faisabilité de la mise en œuvre de l'approche proposée sur des véhicules réalistes. En outre, des simplifications importantes du modèle et des hypothèses sur l'environnement sont prises

en compte dans la plupart des systèmes. Par conséquent, limiter l'application de ces systèmes.

Nous formulons le contrôleur MPC comme un problème d'optimisation de dimension finie. Par conséquent, nous discrétisons la dynamique du véhicule ξ avec un temps d'échantillonnage T_s.

$$\xi(k+1) = f(\xi(k), u(k)) \tag{A.8a}$$
$$u(k+1) = u(k) + \Delta u(k) \tag{A.8b}$$

Où $\xi(k)$ est l'état du vecteur système à l'instant k, et $u(k)$ est le vecteur des variables manipulées à déterminer par le contrôleur.

$$\xi = [\dot{x}, \dot{y}, \psi, \dot{\psi}, e_\psi, e_y] \tag{A.9}$$

Les séquences de contrôle et d'état doivent satisfaire $u(k) \in \mathbb{U}$ et $\xi(k) \in \mathbb{X}$. Où X est l'ensemble admissible de l'état du système et U est l'ensemble admissible de l'entrée de commande et comprend l'angle de braquage, le couple de freinage et l'accélérateur.$f(\xi(k), u(k))$ décrit le système dynamique. Le problème d'optimisation en boucle ouverte de la commande prédictive du modèle à horizon fini à l'instant k est formulé comme suit :

$$\min_{u} \quad J(\boldsymbol{\xi}, \boldsymbol{u}) \tag{A.10a}$$
$$\text{s.t.} \quad \xi(k+1) = f(\xi(k), u(k)), k \in [0, ..., K-1], \tag{A.10b}$$
$$u(k+1) = u(k) + \Delta u(k), \tag{A.10c}$$
$$u(k) \in \mathbb{U}, k \in [0, ..., K-1], \tag{A.10d}$$
$$\xi(k) \in \mathbb{X}, k \in [0, ..., K-1], \tag{A.10e}$$
$$\xi_0 = \xi \tag{A.10f}$$

$\boldsymbol{\xi} = [\xi(0), ..., \xi(K)]$ est la séquence d'états obtenue en appliquant la séquence de contrôle $\boldsymbol{u} = [u(0), .., u(K-1)]$ au système à partir de l'état initial ξ_0.

En plus du suivi de trajectoire, le contrôleur MPC permet également d'éviter les obstacles sur la trajectoire de référence, détectés à l'aide de capteurs lidar. La fonction de coût $J_{\Delta U}$ tient compte à la fois du suivi de la ligne médiane et de l'évitement d'obstacles. Où $\eta = [\dot{x}, \dot{\psi}, \dot{e_\psi}, \dot{e_y}]$ et $\eta = [\dot{x}_{ref}, \dot{\psi}, 0, 0, \dot{\psi}', \dot{x}_{ref}]$.$\eta_k$,$\eta_{ref_t}$ représente l'écart de les états de suivi à partir de la référence au centre de la voie, et Q, R,S sont des matrices de pondération.

$$\begin{aligned} J(\xi, u_t, \Delta u_t) = \sum_{k=t}^{t+H-1} & (||\eta_k - \eta_{ref_t}||^2 Q \\ & -||u_k||^2 R - ||\Delta u_k||^2 S) \end{aligned} \tag{A.11}$$

Afin de s'assurer que la stratégie de commande prédictive est réalisable, le contrôleur doit s'assurer que les contraintes d'accélération du véhicule sont dans les marges possibles ; de plus, la contrainte assure que le véhicule reste en marche avant et ne quitte pas la route.

$$\begin{cases} a_{min} < a < a_{max} \\ a_{min} < a < a_{max} \\ \frac{-\pi}{2} < e_{\psi} < \frac{\pi}{2} \\ -\rho < e_y < \rho \end{cases} \tag{A.12}$$

3.3 Architecture du système

Figure A.10 fournit l'architecture globale du système proposé. Les entrées de la caméra stéréoscopique sont transmises à un modèle Yolov5 pour effectuer la détection d'objets ; ensuite, un modèle DeepSort formé fournira un suivi de l'objet détecté pour le déplacement et l'arrêt. La position et la vitesse relatives des véhicules détectés sont transmises au modèle de commande prédictive du modèle pour effectuer un évitement d'obstacles.

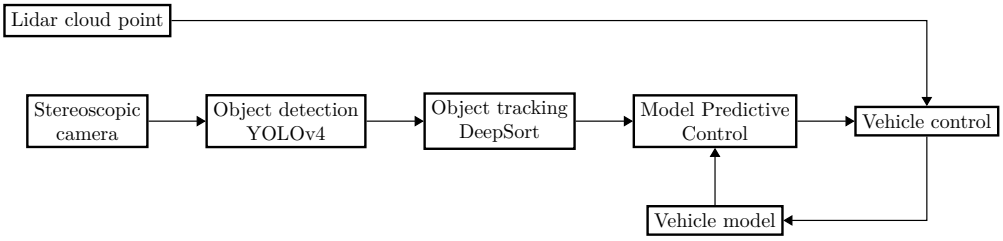


Figure A.10: System architecture.

3.3.1 conclusion

En raison de la nature dangereuse des manœuvres de dépassement et de l'importance d'un dépassement optimal des véhicules pour dégager le trafic et éviter les longs convois, dans cette thèse, nous avons proposé un système permettant de dépasser en toute sécurité un véhicule de tête dans différentes conditions de circulation et architectures routières.

Le système utilise des caméras stéréoscopiques montées à l'avant du véhicule pour détecter l'architecture de la route, telles que la largeur et le centre de la voie, en plus de capteurs lidar pour positionner avec précision les objets mobiles et stationnaires présents sur la route, tels que les autres véhicules et les objets dangereux.

Le système proposé est basé sur l'architecture Model Predictive Control MPC qui offre la possibilité de prédire les états futurs d'un système à partir d'une modélisation de ses composants. Pour modéliser le mouvement du véhicule, nous avons utilisé un modèle de vélo dynamique capable de représenter avec précision l'état futur du véhicule, notamment dans les manœuvres rapides telles que la manœuvre de dépassement. Contrairement aux solutions existantes, qui n'utilisent que des modèles cinématiques simplifiés et supposent une connaissance préalable de l'environnement, les rendant inutilisables dans des scénarios réels.

Pour tester les performances du système proposé, nous avons introduit un ensemble de scénarios qui varient de routes droites à courbes dans différentes conditions météorologiques et de circulation, principalement la présence de brouillard car il a une influence beaucoup plus importante sur les capteurs, en particulier les capteurs de caméra. Et dans tous les scénarios, nous avons remarqué que le système pouvait effectuer un dépassement en toute sécurité tout en respectant les contraintes prédéfinies concernant l'accélération, la vitesse

et le cheminement principalement à une vitesse inférieure. Cependant, les performances du système ont progressivement diminué à mesure que la vitesse des véhicules augmentait. Une autre remarque cruciale est la forte influence du brouillard sur les performances du système.

Nous avons également observé que, comme pour les conducteurs humains, le comportement du véhicule de tête influence également le résultat de la manœuvre, car un changement de vitesse aléatoire au milieu d'un dépassement peut entraîner l'annulation du dépassement ou même des accidents. Par conséquent, naturellement, puisque le dépassement dépend de la coordination entre les deux véhicules, le changement de vitesse inattendu a réduit l'efficacité du MPC commandé, en particulier dans la deuxième étape du dépassement.

Bibliography

[1] Committee, O.-R. A. D. (, "Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles." Jun. 2018.

[2] Lamouik, I., Yahyaouy, A., and Sabri, M. A., "Model predictive control for full autonomous vehicle overtaking," *Transportation Research Record: Journal of the Transportation Research Board*, Jan. 2023.

[3] Lamouik, I., Yahyaouy, A., and Sabri, M. A., "Distributed multi-cluster dynamic q-routing for large size traffic grids," *International Journal of Cloud Computing*, 2022.

[4] Lamouik, I., Yahyaouy, A., and Sabri, M. A., "Smart multi-agent traffic coordinator for autonomous vehicles at intersections," in *2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, IEEE, May 2017, pp. 1–6.

[5] Lamouik, I., Yahyaouy, A., and Sabri, M. A., "Deep neural network dynamic traffic routing system for vehicles," in *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*, IEEE, Apr. 2018, pp. 1–4.

[6] Jiber, M., Lamouik, I., Yahyaouy, A., and Sabri, M. A., "Traffic flow prediction using neural network," in *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*, IEEE, Apr. 2018, pp. 1–4.

[7] Douglass B. Lee, J., Klein, L. A., and Camus, G., "Induced traffic and induced demand," *Transportation Research Record*, vol. 1659, no. 1, pp. 68–75, 1999.

[8] Schaefer, A., "Commuting takes its toll," *Scientific American Mind*, vol. 16, pp. 14–15, Oct. 2005.

[9] Errampalli, M., Senathipathi, V., and Thamban, D., "Effect of congestion on fuel cost and travel time cost on multi-lane highways in india," *International Journal for Traffic and Transport Engineering*, vol. 5, pp. 458–472, Dec. 2015.

[10] Jünger, M., Reinelt, G., and Rinaldi, G., "The traveling salesman problem," *Handbooks in operations research and management science*, vol. 7, pp. 225–330, 1995.

[11] Xin, L., Song, W., Cao, Z., and Zhang, J., "Neurolkh: Combining deep learning model with lin-kernighan-helsgaun heuristic for solving the traveling salesman problem," in *Advances in Neural Information Processing Systems*, Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., Eds., vol. 34, Curran Associates, Inc., 2021, pp. 7472–7483.

[12] Azi, N., Gendreau, M., and Potvin, J.-Y., "An exact algorithm for a single-vehicle routing problem with time windows and multiple routes," *European Journal of Operational Research*, vol. 178, no. 3, pp. 755–766, 2007, ISSN: 0377-2217.

[13] Kim, S. and Moon, I., "Traveling salesman problem with a drone station," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 1, pp. 42–52, 2019.

[14] Ha, Q. M., Deville, Y., Pham, Q. D., and Hà, M. H., "On the min-cost traveling salesman problem with drone," *Transportation Research Part C: Emerging Technologies*, vol. 86, pp. 597–621, 2018, ISSN: 0968-090X.

[15] Agatz, N., Bouman, P., and Schmidt, M., "Optimization approaches for the traveling salesman problem with drone," *Transportation Science*, vol. 52, no. 4, pp. 965–981, 2018.

[16] Pan, B., Zhang, Z., and Lim, A., "Multi-trip time-dependent vehicle routing problem with time windows," *European Journal of Operational Research*, vol. 291, no. 1, pp. 218–231, 2021, ISSN: 0377-2217.

[17] Lu, D. and Gzara, F., "The robust vehicle routing problem with time windows: Solution by branch and price and cut," *European Journal of Operational Research*, vol. 275, no. 3, pp. 925–938, 2019, ISSN: 0377-2217.

[18] Zhang, S., Chen, M., Zhang, W., and Zhuang, X., "Fuzzy optimization model for electric vehicle routing problem with time windows and recharging stations," *Expert Systems with Applications*, vol. 145, p. 113 123, 2020, ISSN: 0957-4174.

[19] Gmira, M., Gendreau, M., Lodi, A., and Potvin, J.-Y., "Tabu search for the time-dependent vehicle routing problem with time windows on a road network," *European Journal of Operational Research*, vol. 288, no. 1, pp. 129–140, 2021, ISSN: 0377-2217.

[20] Tirkolaee, E. B., Abbasian, P., Soltani, M., and Ghaffarian, S. A., "Developing an applied algorithm for multi-trip vehicle routing problem with time windows in urban waste collection: A case study," *Waste Management & Research*, vol. 37, no. 1_suppl, pp. 4–13, 2019.

[21] Naccache, S., Côté, J.-F., and Coelho, L. C., "The multi-pickup and delivery problem with time windows," *European Journal of Operational Research*, vol. 269, no. 1, pp. 353–362, 2018, ISSN: 0377-2217.

[22] Arslan, A. M., Agatz, N., Kroon, L., and Zuidwijk, R., "Crowdsourced delivery–a dynamic pickup and delivery problem with ad hoc drivers," *Transportation Science*, vol. 53, no. 1, pp. 222–235, 2019.

[23] Ulmer, M. W., Thomas, B. W., Campbell, A. M., and Woyak, N., "The restaurant meal delivery problem: Dynamic pickup and delivery with deadlines and random ready times," *Transportation Science*, vol. 55, no. 1, pp. 75–100, 2021.

[24] Danloup, N., Allaoui, H., and Goncalves, G., "A comparison of two meta-heuristics for the pickup and delivery problem with transshipment," *Computers & Operations Research*, vol. 100, pp. 155–171, 2018, ISSN: 0305-0548.

[25] Haddad, M. N., Martinelli, R., Vidal, T., Martins, S., Ochi, L. S., Souza, M. J. F., and Hartl, R., "Large neighborhood-based metaheuristic and branch-and-price for the pickup and delivery problem with split loads," *European Journal of Operational Research*, vol. 270, no. 3, pp. 1014–1027, 2018, ISSN: 0377-2217.

[26] Hornstra, R. P., Silva, A., Roodbergen, K. J., and Coelho, L. C., "The vehicle routing problem with simultaneous pickup and delivery and handling costs," *Computers & Operations Research*, vol. 115, p. 104 858, 2020, ISSN: 0305-0548.

[27] Granada-Echeverri, M., Toro, E., and Santa, J., "A mixed integer linear programming formulation for the vehicle routing problem with backhauls," *International Journal of Industrial Engineering Computations*, vol. 10, no. 2, pp. 295–308, 2019.

[28] Belloso, J., Juan, A. A., and Faulin, J., "An iterative biased-randomized heuristic for the fleet size and mix vehicle-routing problem with backhauls," *International Transactions in Operational Research*, vol. 26, no. 1, pp. 289–301, 2019.

[29] Sethanan, K. and Jamrus, T., "Hybrid differential evolution algorithm and genetic operator for multi-trip vehicle routing problem with backhauls and heterogeneous fleet in the beverage logistics industry," *Computers & Industrial Engineering*, vol. 146, p. 106 571, 2020.

[30] Pearn, W. L. and Wu, T., "Algorithms for the rural postman problem," *Computers & Operations Research*, vol. 22, no. 8, pp. 819–828, 1995.

[31] Fernández, E., Laporte, G., and Rodriguez-Pereira, J., "A branch-and-cut algorithm for the multidepot rural postman problem," *Transportation Science*, vol. 52, no. 2, pp. 353–369, 2018.

[32] Nguyen, L., "An efficient multi-vehicle routing strategy for goods delivery services," 2021.

[33] Zhang, K., He, F., Zhang, Z., Lin, X., and Li, M., "Multi-vehicle routing problems with soft time windows: A multi-agent reinforcement learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 121, p. 102 861, 2020.

[34] Bahalke, U., Hamta, N., Shojaeifard, A. R., Alimoradi, M., and Rabiee, S., "A new heuristic algorithm for multi vehicle routing problem with and/or-type precedence constraints and hard time windows," *Operational Research in Engineering Sciences: Theory and Applications*, vol. 5, no. 2, pp. 28–60, Jun. 2022.

[35] Mansouri, M., Lagriffoul, F., and Pecora, F., "Multi vehicle routing with nonholonomic constraints and dense dynamic obstacles," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 3522–3529.

[36] Gholami-Zanjani, S. M., Jafari-Marandi, R., Pishvaee, M. S., and Klibi, W., "Dynamic vehicle routing problem with cooperative strategy in disaster relief," *International Journal of Shipping and Transport Logistics*, vol. 11, no. 6, pp. 455–475, 2019.

[37] Altabeeb, A. M., Mohsen, A. M., Abualigah, L., and Ghallab, A., "Solving capacitated vehicle routing problem using cooperative firefly algorithm," *Applied Soft Computing*, vol. 108, p. 107 403, 2021, ISSN: 1568-4946.

[38] Altabeeb, A. M., Mohsen, A. M., and Ghallab, A., "An improved hybrid firefly algorithm for capacitated vehicle routing problem," *Applied Soft Computing*, vol. 84, p. 105 728, 2019, ISSN: 1568-4946.

[39] Kır, S., Yazgan, H. R., and Tüncel, E., "A novel heuristic algorithm for capacitated vehicle routing problem," *Journal of Industrial Engineering International*, vol. 13, no. 3, pp. 323–330, 2017.

[40] Fava, L. P., Furtado, J. C., Helfer, G. A., Barbosa, J. L. V., Beko, M., Correia, S. D., and Leithardt, V. R. Q., "A multi-start algorithm for solving the capacitated vehicle routing problem with two-dimensional loading constraints," *Symmetry*, vol. 13, no. 9, 2021, ISSN: 2073-8994.

[41] Zhuang, J., Luo, H., Pan, T.-S., and Pan, J.-S., "Improved flower pollination algorithm for the capacitated vehicle routing problem," *J. Netw. Intell*, vol. 5, no. 3, pp. 141–156, 2020.

[42] Lian, Y., Xie, W., and Zhang, L., "A probabilistic time-constrained based heuristic path planning algorithm in warehouse multi-agv systems.," *IFAC*, vol. 53, no. 2, pp. 2538–2543, 2020, IFAC World Congress, ISSN: 2405-8963.

[43] Sinnl, M., "Mixed-integer programming approaches for the time-constrained maximal covering routing problem," *OR Spectrum*, vol. 43, no. 2, pp. 497–542, 2021.

[44] Wu, Y. and Zhang, Y., "Dynamic time-constrained path recommendation with user's preference," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2021, pp. 1641–1648.

[45] Jotrao, S. and Batta, R., "Time-constrained uav path planning in 3d network for maximum information gain," *Military Operations Research*, vol. 26, no. 3, pp. 5–25, 2021, ISSN: 10825983, 21632758. (visited on 12/25/2022).

[46] Kim, C., Ko, Y.-B., and Vaidya, N. H., "Link-state routing without broadcast storming for multichannel mesh networks," *Computer Networks*, vol. 54, no. 2, pp. 330–340, 2010, Wireless Multi-Hop Networking for Infrastructure Access.

[47] Geng, H., Shi, X., Wang, Z., and Yin, X., "A hop-by-hop dynamic distributed multipath routing mechanism for link state network," *Computer Communications*, vol. 116, pp. 225–239, 2018.

[48] Rahmani, A. M., Ali, S., Yousefpoor, E., Yousefpoor, M. S., Javaheri, D., Lalbakhsh, P., Hassan Ahmed, O., Hosseinzadeh, M., and Lee, S.-W., "Olsr+: A new routing method based on fuzzy logic in flying ad-hoc networks (fanets)," *Vehicular Communications*, vol. 36, p. 100 489, 2022.

[49] Zhang, L., Hu, L., Hu, F., Ye, Z., Li, X., and Kumar, S., "Enhanced olsr routing for airborne networks with multi-beam directional antennas," *Ad Hoc Networks*, vol. 102, p. 102 116, 2020.

[50] Zhang, D.-g., Cui, Y.-y., and Zhang, T., "New quantum-genetic based olsr protocol (qg-olsr) for mobile ad hoc network," *Applied Soft Computing*, vol. 80, pp. 285–296, 2019.

[51] Yi, J., Adnane, A., David, S., and Parrein, B., "Multipath optimized link state routing for mobile ad hoc networks," *Ad Hoc Networks*, vol. 9, no. 1, pp. 28–47, 2011.

[52] Zhang, D.-g., Zhang, T., Dong, Y., Liu, X.-h., Cui, Y.-y., and Zhao, D.-x., "Novel optimized link state routing protocol based on quantum genetic strategy for mobile learning," *Journal of Network and Computer Applications*, vol. 122, pp. 37–49, 2018.

[53] Jubair, M. A., Mostafa, S. A., Muniyandi, R. C., Mahdin, H., Mustapha, A., Hassan, M. H., Mahmoud, M. A., Al-Jawhar, Y. A., Al-Khaleefa, A. S., and Mahmood, A. J., "Bat optimized link state routing protocol for energy-aware mobile ad-hoc networks," *Symmetry*, vol. 11, no. 11, 2019.

[54] Belamri, F., Boulfekhar, S., and Aissani, P. D., "Aclsgo: Ant colony link state aware geographic opportunistic routing protocol for vanet," in *2021 International Conference on Computational Science and Computational Intelligence (CSCI)*, 2021, pp. 1348–1354.

[55] "Routing Information Protocol," RFC 1058, Jun. 1988.

[56] Hu, Y.-C., Johnson, D. B., and Perrig, A., "Sead: Secure efficient distance vector routing for mobile wireless ad hoc networks," *Ad Hoc Networks*, vol. 1, no. 1, pp. 175–192, 2003.

[57] Farooq, A. and Yuksel, M., "Distance vector routing in partitioned networks," in *2022 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, 2022, pp. 1–7.

[58] Gupta, B., Koneru, S., Alyanbaawi, A., Rahimi, N., and Liu, Z., "A modified version of dvr-based multicasting with security," in *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, 2017, pp. 931–936.

[59] He, G., "Destination-sequenced distance vector (dsdv) protocol," *Networking Laboratory, Helsinki University of Technology*, vol. 135, pp. 1–9, 2002.

[60] Abdel Jawad, M., Salah, S., and Zaghal, R., "Dsdv extension to enhance the performance of ad hoc networks in high diverse-velocity environments," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 14, no. 06, pp. 95–114, Apr. 2020.

[61] Khan, K. U. R., Zaman, R. U., Reddy, A. V., Reddy, K. A., and Harsha, T. S., "An efficient dsdv routing protocol for wireless mobile ad hoc networks and its performance comparison," in *2008 Second UKSIM European Symposium on Computer Modeling and Simulation*, 2008, pp. 506–511.

[62] Fengjie, Y., Hui, Y., and Ying, Z., "Research on dsdv routing protocol based on wireless mesh network," in *2018 Chinese Control And Decision Conference (CCDC)*, 2018, pp. 4292–4297.

[63] Perkins, C., Belding-Royer, E., and Das, S., "Rfc3561: Ad hoc on-demand distance vector (aodv) routing," 2003.

[64] Clausen, T., Yi, J., and Herberg, U., "Lightweight on-demand ad hoc distance-vector routing - next generation (loadng): Protocol, extension, and applicability," *Computer Networks*, vol. 126, pp. 125–140, 2017.

[65] Tavizi, A. and Ghaffari, A., "Tree-based reliable and energy-aware multicast routing protocol for mobile ad hoc networks," *The Journal of Supercomputing*, vol. 74, Nov. 2018.

[66] Chavan, C. P. and Venkataram, P., "Design and implementation of event-based multicast aodv routing protocol for ubiquitous network," *Array*, vol. 14, p. 100 129, 2022.

[67] C.R, R. and Periasamy, P., "Hybrid mobile ad-hoc delay tolerant network for optimum routing in wireless sensor networks," vol. 08, pp. 1303–1308, Sep. 2019.

[68] Chen, T.-W. and Gerla, M., "Global state routing: A new routing scheme for ad-hoc wireless networks," in *ICC'98. 1998 IEEE International Conference on Communications. Conference Record. Affiliated with SUPERCOMM'98 (Cat. No. 98CH36220)*, IEEE, vol. 1, 1998, pp. 171–175.

[69] Jing, C., Hua, C. G., and Liang, H., "A secure global state routing for mobile ad hoc networks," in *Personal Wireless Communications*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 228–238.

[70] Ivanescu, T., El-Hajjar, M., Merrett, G., and Yetgin, H., "Route lifetime analysis for vehicular networks," in *2021 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS): W5 :AI/Machine Learning Enabled Connected Vehicles (13/12/21)*, Nov. 2021.

[71] Pei, G., Gerla, M., and Chen, T.-W., "Fisheye state routing: A routing scheme for ad hoc wireless networks," in *2000 IEEE International Conference on Communications. ICC 2000. Global Convergence Through Communications. Conference Record*, vol. 1, 2000, 70–74 vol.1.

[72] Hassan, M. A., Ullah, S. I., Khan, I. U., Hussain Shah, S. B., Salam, A., and Ullah Khan, A. W., "Unmanned aerial vehicles routing formation using fisheye state routing for flying ad-hoc networks," in *The 4th International Conference on Future Networks and Distributed Systems (ICFNDS)*, ser. ICFNDS '20, St.Petersburg, Russian Federation: Association for Computing Machinery, 2020.

[73] Alilou, M. and Hatamlou, A., "A novel routing algorithm for mobile ad-hoc networks based on q-learning and its generalization to fsr routing protocol," *Computer and Knowledge Engineering*, vol. 1, no. 2, pp. 27–32, 2018.

[74] Akbar, R. Z., Istikmal, and Sussi, "Performance analysis fsr and dsr routing protocol in vanet with v2v and v2i models," in *2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 2020, pp. 158–163.

[75] Chiang, C.-C., Wu, H.-K., Liu, W., and Gerla, M., "Routing in clustered multihop, mobile wireless networks with fading channel," in *proceedings of IEEE SICON*, Citeseer, vol. 97, 1997, pp. 197–211.

[76] Devarajan, K. and Padmathilagam, V., "An enhanced cluster gateway switch routing protocol (ecgsr) for congestion control using aodv algorithm in manet," *International Journal of Computer Applications*, vol. 123, no. 3, 2015.

[77] Sahaya Sheela, M. A. and Prabakaran, R., "Improvement of battery lifetime in software-defined network using particle swarm optimization based cluster-head gateway switch routing protocol with fuzzy rules," *Computational Intelligence*, vol. 36, no. 2, pp. 813–823, 2020.

[78] Garg, P., Gupta, M., Joon, R., and Gupta, S., "A novel approach for malicious node detection in cluster-head gateway switching routing in mobile ad hoc networks," *International Journal of Future Generation Communication and Networking*, vol. 13, pp. 99–111, Aug. 2020.

[79] Murthy, S. and Garcia-Luna-Aceves, J., "An efficient routing protocol for wireless networks," *Mobile Networks and Applications*, vol. 1, Feb. 1970.

[80] Hu, Y.-C., Maltz, D. A., and Johnson, D. B., "The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4," RFC 4728, Feb. 2007.

[81] Shen, Z., Wang, W., Shen, Q., Zhu, S., Fardoun, H. M., and Lou, J., "A novel learning method for multi-intersections aware traffic flow forecasting," *Neurocomputing*, vol. 398, no. 2020, pp. 477–484, 2020.

[82] Wang, S., Huang, W., and Lo, H. K., "Traffic parameters estimation for signalized intersections based on combined shockwave analysis and bayesian network," *Transp. Res. Part C: Emerg. Technol.*, vol. 104, no. 2019, pp. 22–37, Jul. 2019.

[83] Patil, S. S. and Soma, S., "Traffic congestion Re-Routing control system using fuzzy logic," in *International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018*, ser. Lecture Notes on Data Engineering and Communications Technologies, vol. 26, Cham: Springer International Publishing, 2019, pp. 778–785.

[84] Neumann, T., "Fuzzy routing algorithm in telematics transportation systems," in *Smart Solutions in Today's Transport*, ser. Communications in Computer and Information Science, vol. 715, Cham: Springer International Publishing, 2017, pp. 494–505.

[85] Chen, S., Yang, J., Li, Y., and Yang, J., "Multiconstrained network intensive vehicle routing adaptive ant colony algorithm in the context of neural network analysis," en, *Complexity*, vol. 2017, no. Special Issue, pp. 1–9, Sep. 2017.

[86] Delling, D., Sanders, P., Schultes, D., and Wagner, D., "Engineering route planning algorithms," in *Algorithmics of Large and Complex Networks: Design, Analysis, and Simulation*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 117–139.

[87] Keren, D., Yasar, A.-U.-H., Knapen, L., Cho, S., Bellemans, T., Janssens, D., Wets, G., Schuster, A., and Sharfman, I., "Exploiting graph-theoretic tools for matching and partitioning of agent population in an agent-based model for traffic and transportation applications," *Procedia Comput. Sci.*, vol. 10, pp. 833–839, Jan. 2012.

[88] Delling, D., Goldberg, A. V., Razenshteyn, I., and Werneck, R. F., "Graph partitioning with natural cuts," in *2011 IEEE International Parallel Distributed Processing Symposium*, May 2011, pp. 1135–1146.

[89] Karypis, G., Han, E.-H., and Kumar, V., "Chameleon a hierarchical clustering algorithm using dynamic modeling," *Computer*, vol. 32, pp. 68–75, Sep. 1999.

[90] Sanders, P. and Schulz, C., "Distributed evolutionary graph partitioning," in *2012 Proceedings of the Meeting on Algorithm Engineering and Experiments (ALENEX)*, ser. Proceedings, Society for Industrial and Applied Mathematics, Jan. 2012, pp. 16–29.

[91] Walshaw, C. and Cross, M., "Jostle: Parallel multilevel graph-partitioning software - an overview," *Mesh Partitioning Techniques and Domain Decomposition Techniques*, Jan. 2007.

[92] Dijkstra, E. W., "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[93] Brooks, R. A. and Lozano-Pérez, T., "A subdivision algorithm in configuration space for findpath with rotation," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-15, no. 2, pp. 224–233, Mar. 1985.

[94] Lee, C. Y., "An algorithm for path connections and its applications," *IRE Transactions on Electronic Computers*, vol. EC-10, no. 3, pp. 346–365, Sep. 1961.

[95] Correll, N., "Introduction to Autonomous Robots," 2nd. Magellan Scientific, 2014.

[96] Dinitz, Y. and Itzhak, R., "Hybrid bellman-ford-dijkstra algorithm," *Journal of Discrete Algorithms*, vol. 42, no. Supplement C, pp. 35–44, 2017.

[97] Alexopoulos, C. and Griffin, P. M., "Path planning for a mobile robot," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 2, pp. 318–322, Mar. 1992.

[98] Majd, Y. A. E., Ghazi, H. E., and Nahhal, T., "Parrt: Parallel rapidly exploring random tree (rrt) based on mapreduce," in *2017 International Conference on Electrical and Information Technologies*, Nov. 2017, pp. 1–5.

[99] Li, M., Wang, C., Chen, Z., Lu, X., Wu, M., and Hou, P., "Path planning of mobile robot based on genetic algorithm and gene rearrangement," in *2017 Chinese Automation Congress (CAC)*, Oct. 2017, pp. 6999–7004.

[100] Santiago, R. M. C., Ocampo, A. L. D., Ubando, A. T., Bandala, A. A., and Dadios, E. P., "Path planning for mobile robots using genetic algorithm and probabilistic roadmap," Dec. 2017, pp. 1–5.

[101] Yang, L., Lin, J., Wang, D., and Jia, L., "Dynamic route guidance algorithm based on artificial immune system," *Journal of Control Theory and Applications*, vol. 5, no. 4, pp. 385–390, Nov. 2007.

[102] Xu, Q. L., Cai, M. M., and Zhao, L. H., "The robot path planning based on ant colony and particle swarm fusion algorithm," in *2017 Chinese Automation Congress (CAC)*, Oct. 2017, pp. 411–415.

[103] Yu, K. M., Lee, M. G., and Chi, S. S., "Dynamic path planning based on adaptable ant colony optimization algorithm," in *2017 Sixth International Conference on Future Generation Communication Technologies (FGCT)*, Aug. 2017, pp. 1–7.

[104] Savelsbergh, M. W. P., "Local search in routing problems with time windows," *Annals of Operations Research*, vol. 4, no. 1, pp. 285–305, Dec. 1985.

[105] Gao, K., Zhang, Y., Sadollah, A., and Su, R., "Optimizing urban traffic light scheduling problem using harmony search with ensemble of local search," *Applied Soft Computing*, vol. 48, pp. 359–372, Nov. 2016.

[106] Sumner, R., "Cell messaging process for an in-vehicle traffic congestion information system," US Patent 5,182,555, Jan. 1993.

[107] Jin, Q., Wu, G., Boriboonsomsin, K., and Barth, M., "Multi-agent intersection management for connected vehicles using an optimal scheduling approach," *IEEE*, vol. 10.1109/ICCVE.2012.41, pp. 185–190, 2012.

[108] Silberschatz, A. and Galvin Peter B.and Gagne, G., "Operating System Concepts," eighth, Round Robin Scheduling.

[109] Tindell, K. and Hansson, H., "Real time systems by fixed priority scheduling," M.S. thesis, Department of Computer Systems, Uppsala University, 1997.

[110] Behera, H. S., Naik, R. K., and Parida, S., "Improved multilevel feedback queue scheduling using dynamic time quantum and its performance analysis," *International Journal of Computer Science and Information Technologies,*, vol. 3, pp. 3801–3807, 2012.

[111] Dresner, K. and Stone, P., "A multiagent approach to autonomous intersection management," *Journal of Artificial Intelligence Research*, vol. 31, pp. 591–656, Mar. 2008.

[112] Kamal, M. A. S., Imura, J. i., Hayakawa, T., Ohata, A., and Aihara, K., "A vehicle-intersection coordination scheme for smooth flows of traffic without using traffic lights," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1136–1147, Jun. 2015.

[113] Pérez, J., Milanés, V., Onieva, E., Godoy, J., and Alonso, J., "Longitudinal fuzzy control for autonomous overtaking," in *2011 IEEE International Conference on Mechatronics*, 2011, pp. 188–193.

[114] Naranjo, J. E., Gonzalez, C., Garcia, R., and De Pedro, T., "Lane-change fuzzy control in autonomous vehicles for the overtaking maneuver," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 3, pp. 438–450, 2008.

[115] Mo, S., Pei, X., and Chen, Z., "Decision-making for oncoming traffic overtaking scenario using double dqn," in *2019 3rd Conference on Vehicle Control and Intelligence (CVCI)*, 2019, pp. 1–4.

[116] Chen, W., Xie, G., Ji, W., Fei, R., Hei, X., Li, S., and Ma, J., "Decision making for overtaking of unmanned vehicle based on deep q-learning," in *2021 IEEE 10th Data Driven Control and Learning Systems Conference (DDCLS)*, 2021, pp. 350–353.

[117] Li, X., Xu, X., and Zuo, L., "Reinforcement learning based overtaking decision-making for highway autonomous driving," in *2015 Sixth International Conference on Intelligent Control and Information Processing (ICICIP)*, 2015, pp. 336–342.

[118] Zheng, R., Liu, C., and Guo, Q., "A decision-making method for autonomous vehicles based on simulation and reinforcement learning," in *2013 International Conference on Machine Learning and Cybernetics*, vol. 01, 2013, pp. 362–369.

[119] Kebriaei, H., Rahimi-Kian, A., and Nili Ahmadabadi, M., "Model-based and learning-based decision making in incomplete information cournot games: A state estimation approach," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 4, pp. 713–718, 2015.

[120] Wu, L.-T., Tran, V. L., and Lin, H.-Y., "Real-time overtaking vehicle detection based on optical flow and convolutional neural network," in *Smart Cities, Green Technologies and Intelligent Transport Systems*, Cham: Springer, 2018, pp. 227–243.

[121] Kenji, N., Eisuke, A., and Takio, K., "The recognition of a safe-overtaking driving environment," *IFAC Proceedings*, vol. 40, no. 16, pp. 431–436, 2007.

[122] Junxian Wang, Bebis, G., and Miller, R., "Overtaking vehicle detection using dynamic and quasi-static background modeling," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, 2005, pp. 64–64.

[123] Song, H., Liang, H., Li, H., Dai, Z., and Yun, X., "Vision-based vehicle detection and counting system using deep learning in highway scenes," *European Transport Research Review*, vol. 11, no. 1, p. 51, 2019.

[124] Dimililer, K., Ever, Y. K., and Mustafa, S. M., "Vehicle detection and tracking using machine learning techniques," in *International Conference on Theory and Application of Soft Computing, Computing with Words and Perceptions*, Springer, 2019, pp. 373–381.

[125] Dan, Z. and Jian, W., "Radar-based target identification and tracking on a curved road," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of automobile engineering*, vol. 226, no. 1, pp. 39–47, 2012.

[126] Glaser, S., Vanholme, B., Mammar, S., Gruyer, D., and Nouvelière, L., "Maneuver-based trajectory planning for highly autonomous vehicles on real road with traffic and driver interaction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 589–606, 2010.

[127] Mohamed, Y. and Ahmed, H., "Dual-mode forward collision avoidance algorithm based on vehicle-to-vehicle communication," in *International Midwest Symposium on Circuits and Systems*, 2018, pp. 739–742.

[128] Mo, C., Li, Y., and Zheng, L., "Simulation and analysis on overtaking safety assistance system based on vehicle-to-vehicle communication," *Automotive Innovation*, vol. 1, no. 2, pp. 158–166, Apr. 2018.

[129] Michael, M. e. a., "Vehicular ad-hoc network simulations of overtaking maneuvers on two-lane rural highways," *Transportation research part C: emerging technologies*, vol. 72, pp. 60–76, 2016.

[130] Andrews, S., "Vehicle-to-vehicle (v2v) and vehicle-to-infrastructure (v2i) communications and cooperative driving," in *Handbook of Intelligent Vehicles*. London: Springer London, 2012, pp. 1121–1144.

[131] Murgovski, N. and Sjöberg, J., "Predictive cruise control with autonomous overtaking," in *2015 54th IEEE Conference on Decision and Control (CDC)*, 2015, pp. 644–649.

[132] Andrew, G., Yiqi, G., Theresa, L., J. Karl, H., Eric Tseng, H., and Francesco, B., "Predictive control for agile semi-autonomous ground vehicles using motion primitives," in *American Control Conference*, 2012, pp. 4239–4244.

[133] Qian, X., "Model predictive control for autonomous and cooperative driving," Ph.D. dissertation, PSL Research University, France, Dec. 2016.

[134] Julia, N., Mohammad, A., Paolo, F., and Jonas, S., "Predictive manoeuvre generation for automated driving," in *16th International IEEE Conference on Intelligent Transportation Systems*, 2013, pp. 418–423.

[135] Fenghui, W., Ming, Y., and Ruqing, Y., "Conflict-probability-estimation-based overtaking for intelligent vehicles," *Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 366–370, 2009.

[136] Nguyen, N. A., Moser, D., Schrangl, P., del Re, L., and Jones, S., "Autonomous overtaking using stochastic model predictive control," in *11th Asian Control Conference*, 2017, pp. 1005–1010.

[137] Fatemeh, M., Sergii, V., and Erik, F., "Deep learning model predictive control for autonomous driving in unknown environments," *IFAC*, vol. 51, no. 22, pp. 447–452, 2018, Symposium on Robot Control SYROCO 2018.

[138] Jinsoo, Y., Seongjin, L., Wontaek, L., and Myoungho, S., "Human-like decision-making system for overtaking stationary vehicles based on traffic scene interpretation," *Sensors*, vol. 22, no. 22, 2021, ISSN: 1424-8220.

[139] Alexandru, C., "Designing safety critical software systems to manage inherent uncertainty," in *International Conference on Software Architecture Companion*, 2019, pp. 246–249.

[140] Trieu, V. M., Reza, M., Jindrich, C., and Jaroslav, H., "Model predictive control for autonomous driving vehicles," *Electronics*, vol. 10, no. 21, 2021.

[141] Stone, P. and Veloso, M., "Multiagent systems: A survey from a machine learning perspective," *Autonomous Robots*, vol. 8, no. 3, pp. 345–383, 2000.

[142] Jafari, R. and Javidi, M., "Solving the protein folding problem in hydrophobic-polar model using deep reinforcement learning," *SN Applied Sciences*, vol. 2, Jan. 2020.

[143] Watkins, C. J. and Dayan, P., "Technical note: Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, 1992.

[144] Watkins, C. J. C. H. and Dayan, P., "Q-learning," in *Machine Learning*, 1992, pp. 279–292.

[145] Raman, A., Amitabh, B., Poorya, M., and Anirbit, M., "Understanding deep neural networks with rectified linear units," *CoRR*, 2016.

[146] Samuel, C. and Dit-yan, Y., "Predictive q-routing: A memory-based reinforcement learning approach to adaptive traffic control," in *Advances in Neural Information Processing Systems 8*, Denver, Colorado: MIT Press, 1995, pp. 945–951.

[147] Shi Jianbo and Malik, J., "Normalized cuts and image segmentation," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[148] E, R. B. and E, S. D., "Applied dynamic programming." Princeton university press, 2015, vol. 2050.

[149] contributors, O., "Planet dump retrieved from https://planet.osm.org," 2017.

[150] Gao, Y., "Model Predictive Control for Autonomous and Semiautonomous Vehicles." University of California, Berkeley, 2014.

[151] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A., "You only look once: Unified, real-time object detection," 2016. eprint: 1506.02640.

[152] Redmon, J. and Farhadi, A., "Yolo9000: Better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271. eprint: 1612.08242.

[153] Joseph, R. and Ali, F., "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[154] Alexey, B., Wang, and Liao, C.-Y. M. H.-Y., "Yolov4: Optimal speed and accuracy of object detection," 2020.

[155] Ab-Rahman, A., Sheikh, U., Maliki, M., Heriansyah, R., Singh, K., and Abu-Bakar, S., "Vestro: Velocity estimation using stereoscopic vision," in *2005 1st International Conference on Computers, Communications, & Signal Processing with Special Track on Biomedical Engineering*, IEEE, Dec. 2005, pp. 120–124.

[156] Cehovin, L., Leonardis, A., and Kristan, M., "Visual object tracking performance measures revisited," *IEEE Transactions on Image Processing*, pp. 1–1, 2016.

[157] Petrov, P. and Nashashibi, F., "Modeling and nonlinear adaptive control for autonomous vehicle overtaking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1643–1656, 2014.

[158] Rajamani, R., "Lateral vehicle dynamics," in *Vehicle Dynamics and control*, Springer, 2012, pp. 15–46.

[159] Dixit, S., Fallah, S., Montanaro, U., Dianati, M., Stevens, A., Mccullough, F., and Mouzakitis, A., "Trajectory planning and tracking for autonomous overtaking: State-of-the-art and future prospects," *Annual Reviews in Control*, vol. 45, pp. 76–86, 2018, ISSN: 1367-5788.